CURSO AUTODIDACTICO

INFORMATICA

BASIC

MSX™

con Evaluaciones Periódicas
Diploma del Curso
Autorizado por M.E. y C.

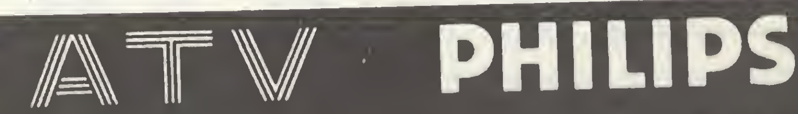
ATV — PHILIPS

JAVIER MURIANO MARTIN

CURSO AUTODIDACTICO

INFORMATICA BASIC

MSX™



IMPRESO EN ESPAÑA
PRINTED IN SPAIN

ISBN: 398 - 5167 - 7

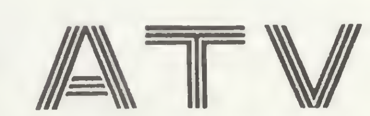
Depósito Legal: MA 985 - 1985

Portada y Montaje JOAQUIN ARIAS
Ilustraciones: J. F. AGUILAR
Imprime: GRAFICAS URANIA
Composición: COMPOSYTEX

CURSO AUTODIDACTICO

INFORMATICA BASIC MSX™





Paseo de la Farola, 25
Telf. 22 81 79
29016 - MALAGA

el curso ATV de Informática-Basic MSX ha sido realizado por:
EQUIPO ATV.
Juan A. Ferrández
Juan F. Rojas
Antonio Fernández
José L. Poveda
Juan F. Aguilar
Voz
Francisco de Linares
Dirección:
Salvador Segura Ruiz

Presentación

El Curso **A.T.V.** de informática y MSX, es un curso de **Enseñanza Audio-Tacto-Visual.** Con estas tres últimas palabras se pretende definir el modo de acceso al alumno, y de comunicación, que se imparte en los cursos A.T.V., y en este caso en el de Informática y Basic MSX

El aprendizaje es una tarea para la que todos los seres vivos están capacitados. **A.T.V.**, teniendo en cuenta las tres vías más importantes de acceso de la información a nuestro cerebro: vista, oído y tacto, parte del supuesto de que un curso de enseñanza audiovisual y de autoestudio, conseguirá el máximo rendimiento en el aprendizaje, fomentando la **participación** del alumno en la tarea.

La informática es una parcela "misteriosa" del conocimiento, que se ha dado en considerar como novedosa, y que se ve envuelta en un halo de hermetismo, solo apto para iniciados. **A.T.V.**, consciente de que ese halo ha envuelto cualquier novedad que se haya producido históricamente en el mundo del conocimiento, de la técnica, pretende, mediante un desenfado no exento de rigor académico, dar a la informática, y al ordenador, el valor de una **herramienta** que como la máquina de escribir o el martillo, cualquiera puede manejar, estando los resultados en función del interés personal, la **curiosidad**, la motivación.

Desprovista la Informática de su carácter de mito moderno, convertida en la herramienta que "**sirve para**", no resulta difícil comprender que **el ordenador no es más que una máquina que hay que manejar con pericia**, y que la **informática no es más que la ciencia que prepara los ordenadores para ser utilizados.**

Para manejar un martillo no es imprescindible conocer las teorías de NEWTON. . . aunque tampoco estorbarían.

Por último partiendo de la base de que **cualquiera está capacitado para aprender Informática**, para manejar un ordenador, puesto que el tratamiento de datos es tan antiguo como la vida, y teniendo en cuenta que el ordenador es una copia de la estructura lógica del cerebro, A.T.V. hace hincapié en esa cualidad: el tratamiento de datos no es una novedad, sí lo es por medio de ordenadores, por eso, y en definitiva, **el curso A.T.V. de Informática y Basic MSX pretende que el alumno comprenda la relación entre Informática, lógica y ordenador.** Nada más.

En cuanto a la mecánica del curso, se compone éste de 12 cintas magnéticas, cada una de las cuales contiene 2 lecciones: Cara A, compuesta de lecciones audiovisuales animadas; y la Cara B conteniendo lecciones prácticcas. En total son 24 lecciones.

Las cintas son la parte audivisual del curso. **Contienen la voz del profesor con las explicaciones y nociones teóricas, que se van ilustrando por medio de imágenes, de modo sincronizado y automático.** El alumno asiste a

cada clase como si estuviera viendo un programa de televisión, en el que se le diera la oportunidad de participar por medio del teclado, siempre bajo las instrucciones del profesor "Bitillo", cicerone en el viaje hacia el universo de la informática.

A lo largo de cada lección el alumno habrá de resolver **múltiples tareas y ejercicios que reforzarán su aprendizaje mediante una retroalimentación continuada**, que le permitirá una asimilación de contenidos tendentes a evitar inútiles memorizaciones. (El hecho de utilizar alternativamente cinta magnética y Libro hace que el alumno aprenda conceptos y técnicas que ya "le suenan" porque han sido esbozados en algunos de los dos instrumentos. De este modo se facilita su labor de estudio).

En el libro se diversifican los ejercicios, teóricos, de repaso, dibujos, diseño de programas, etc., puesto que lo que se pretende en todo momento es enriquecer al alumno con su propia participación.

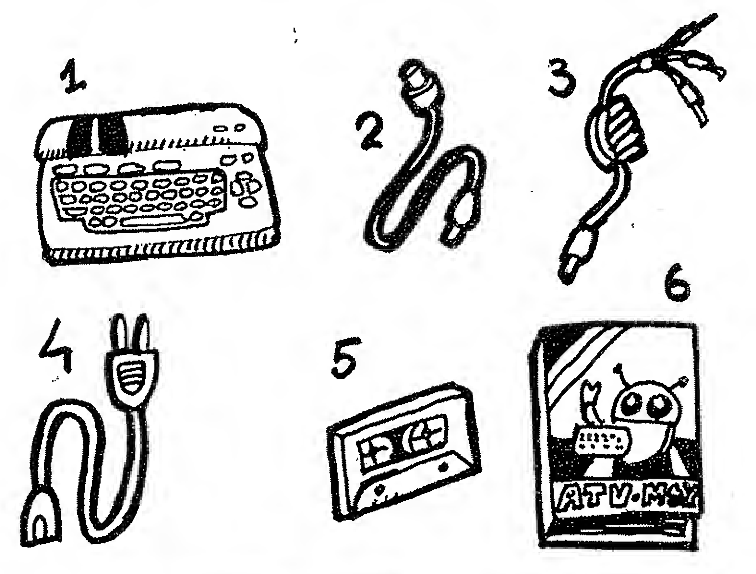
No en vano afirma la antropología moderna que el movimiento de la mano tuvo mucho que ver en el desarrollo del cerebro del hombre. . .

En cuanto al índice de las lecciones del curso, ¡TIENES QUE CONFECCIONARLO TU!

Lección 0

CONTENIDO DEL PAQUETE

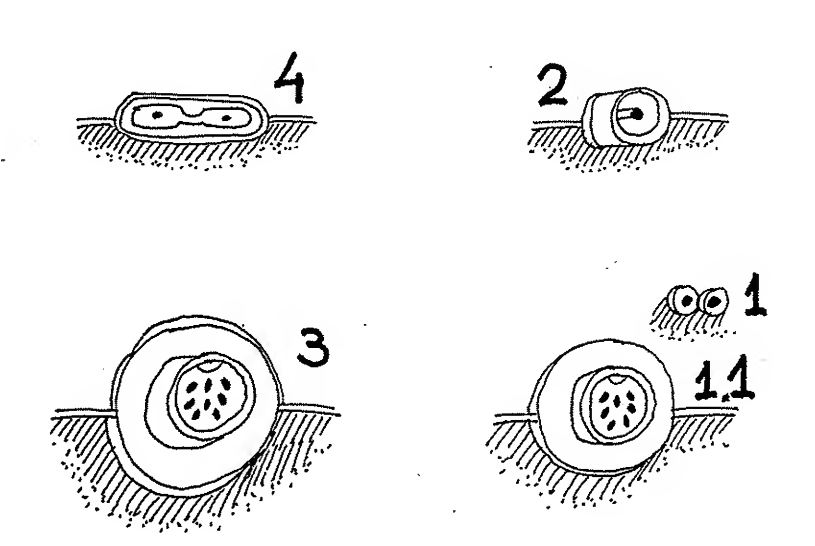
1.– Ordenador MSX
2.– Cable para la pantalla.
3.– Cable para el caset.
4.– Cable de alimentación.
5.– 12 cintas-caset, conteniendo: por la cara A, las lecciones audiovisuales animadas, y por la cara B, las lecciones de práctica.
6.– Libro ATV de Informática y Basic MSX.



CONEXIONES

Tu MSX ha de ser conectado a otros aparatos para que pueda funcionar. Las bocas de entrada de tu ordenador son:

1– Bocas de conexión para monitor. (1.1 NORMA DIN).
2– Boca de conexión al televisor.
3– Boca de conexión al caset.
4– Boca de alimentación o conexión a la red.



Las demás bocas corresponden a los JOYSTICKS, IMPRESORA, UNIDAD DE DISCO, que no vamos a utilizar en nuestro curso. (Si deseas información a este respecto, consulta el manual de referencia de tu MSX).

Vamos a Conectar:

1– Toma el cable del televisor (2) (si tienes monitor, toma el cable del monitor).

2– Coloca un extremo del cable (el macho) en la boca de antena UHF de tu televisor (si tienes monitor, conectarás el cable de monitor).

3– Toma el cable del caset (3), y conecta caset y ordenador. La boca del caset la conocerás por alguno de estos símbolos: TAPE, REC, CASSETTE, ⊙⊙.

De las tres clavijas de que dispone tu cable de caset introducirás en la boca EAR o LINE OUT de tu caset la clavija blanca (si la otra es negra), o la negra (si la otra es roja). La clavija REM, no hay que conectarla para hacer nuestro curso.

Bien, ya has conectado la pantalla y el caset a tu MSX, ahora, en último lugar:

4– Toma el cable de alimentación (4), y conecta el ordenador a la red eléctrica.

5– Enchufa la pantalla (o monitor) y el caset a la red eléctrica.

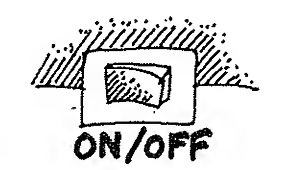
SINTONIZACION

Ya hemos terminado las operaciones de conexión, vamos ahora a sintonizar con TELE **A. T. V.**

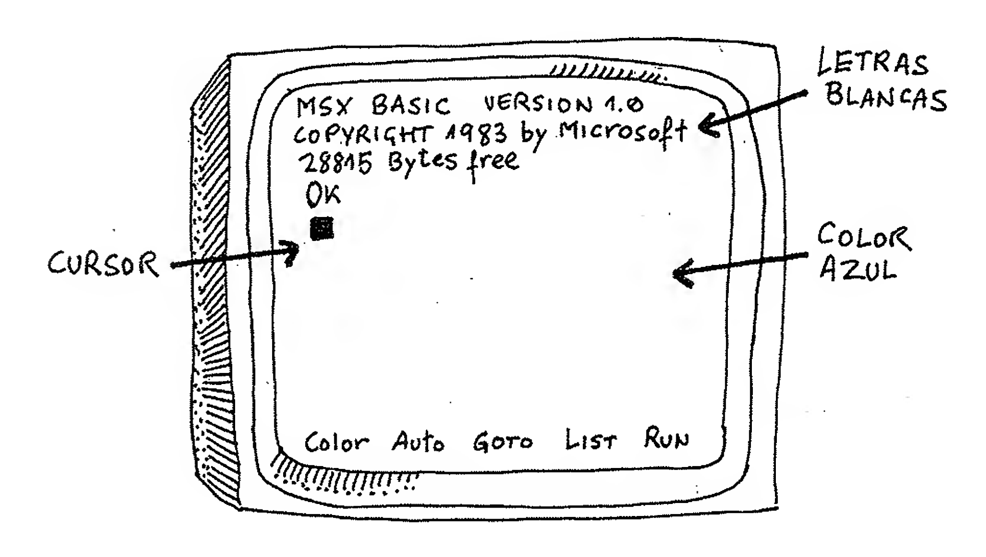
Sí. El ordenador se va a comunicar contigo a través de la pantalla, y es que el ordenador es como una emisora de televisión, y como todas las emisoras, emite por un canal. Así que:

1– Enciende tu televisor.

2– Enciende el ordenador, pulsando el interruptor de encendido ON / OFF, que te comunicará que está activado mediante la lucecita de POWER.



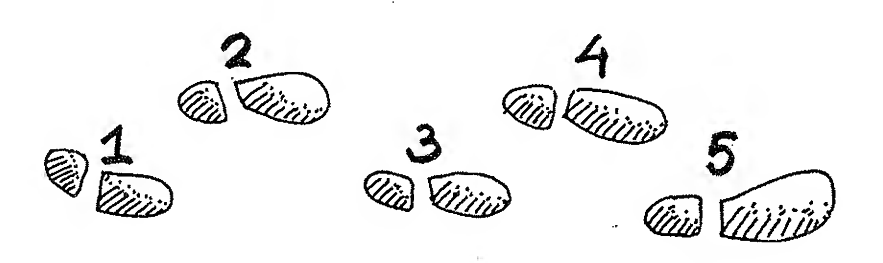
3– Baja el volumen del televisor al mínimo, y utilizando los mandos de sintonización de un canal de tu televisor, no cejes hasta que en tu pantalla aparezca con nitidez esta imagen:



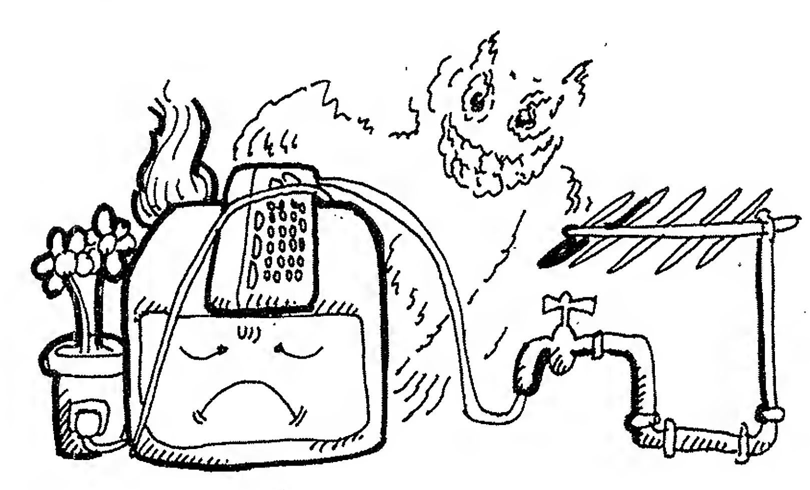
4.– Una vez que hayas localizado esta imagen, sube un poco el volumen del televisor, hasta menos de la mitad, y sigue sintonizando hasta que no se oiga ruido de fondo molesto.

LA CARGA

ATENCION esta operación es sumamente importante, deberás seguir, para hacerlo bien, mis instrucciones paso a paso.



1– Verifica que ordenador, pantalla y caset están bien interconectados, y encendidos.



2– Toma la cinta que contiene la lección 1ª del curso.

3– Sitúa el volumen de tu caset al máximo.



: si tu caset es estéreo, tendrás dificultades para cargar.

(El caset ideal para hacer el curso es uno de calidad media).

4– Sitúa los mandos de graves y agudos, y balance, en la mitad.

5– Introduce la cinta por la cara A.

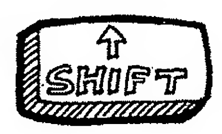
6– Pulsa la tecla de REWIND o REWARD de tu caset, por si la cinta no está rebobinada.

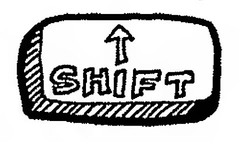
7– Apaga y enciende tu ordenador un par de veces con el interruptor de ON/OFF.

8– Teclea en tu ordenador:

BLOAD "ATV", R

Todas las letras tienen que ir en mayúsculas. Por si no lo sabes, las teclas de mayúsculas están marcadas con una flecha. Aunque también pueden aparecer con su nombre en inglés: SHIFT.







: tiene que ser, exactamente:

BLOAD "ATV", R

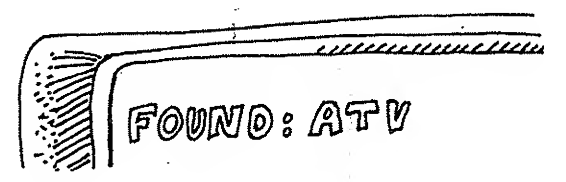
RETURN

Si te equivocas puedes corregir con BS, o apagar y encender nuevamente el ordenador.

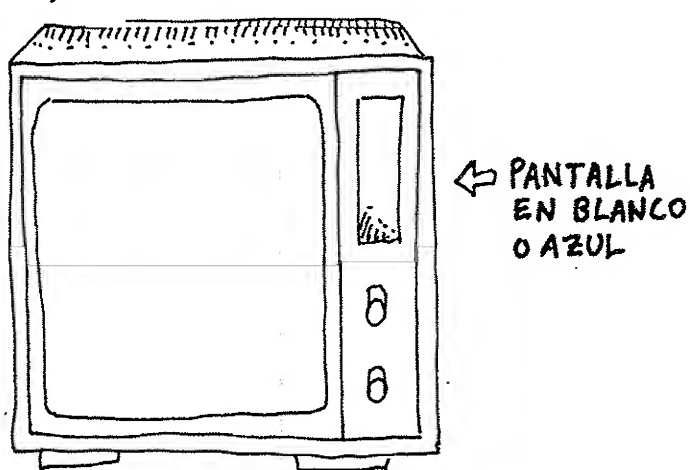
9– Pulsa el PLAY del caset.

HA COMENZADO LA CARGA

10– 30 segundos (más o menos,después de haber pulsado PLAY aparecerá en tu pantalla:



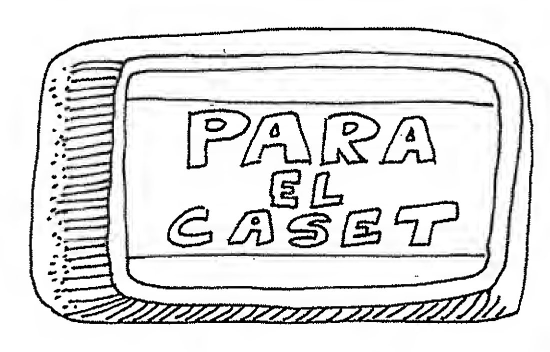
5 segundos (más o menos) después la pantalla se quedará en blanco.(Si tu pantalla es de color, se quedará en azul).



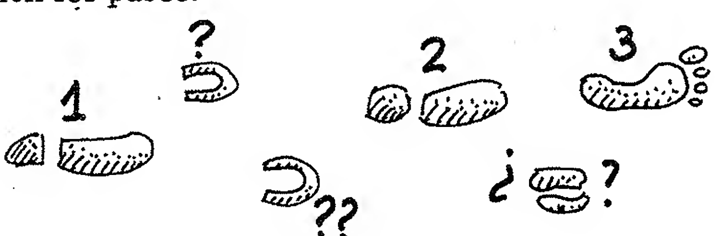
60 segundos (más o menos) después, aparecerá la pantalla de carga en tu televisor:



A partir de este momento tendrás que esperar unos cuatro o cinco minutos hasta que te aparezca esta pantalla.



11– En ese momento, **PARA EL CASET** : ha terminado la CARGA. Empieza la lección. Todos los días - tendrás que hacerlo así, de modo que, apréndete bien los pasos.



PROBLEMAS DE CARGA

La **CARGA** es la operación mediante la cual traspasamos las lecciones, que están grabadas en la cinta-caset, a la memoria del ordenador. (El ordenador tiene una pieza, que se llama MEMORIA, y que sirve para lo mismo que la memoria de las personas).

Si se cumplen los once pasos del epígrafe anterior, no habrá problemas. Si algunos de ellos no se cumple. . .



es que algo ha fallado. Por tanto, **tendrás que empezar desde el principio.**

Errores más comunes

1– Que hayas tecleado mal BLOAD "ATV", R



2– Que tu caset tenga la cabeza sucia.



3– Que el volumen del caset esté muy alto. (Hay que bajarlo un poco).

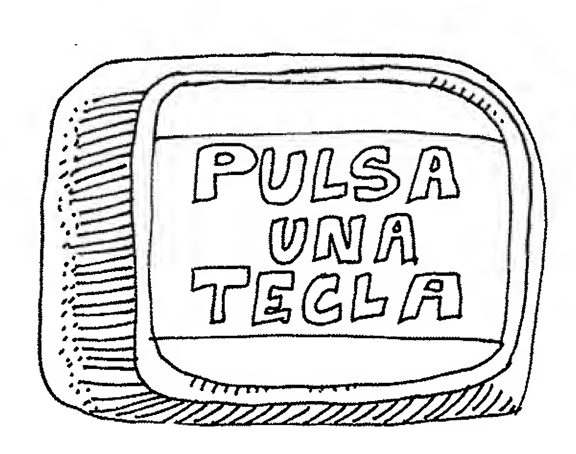


4– Se te ha olvidado encender el ordenador, o alguno otro de los aparatos, o no has rebobinado la cinta, o no has pulsado PLAY. . .



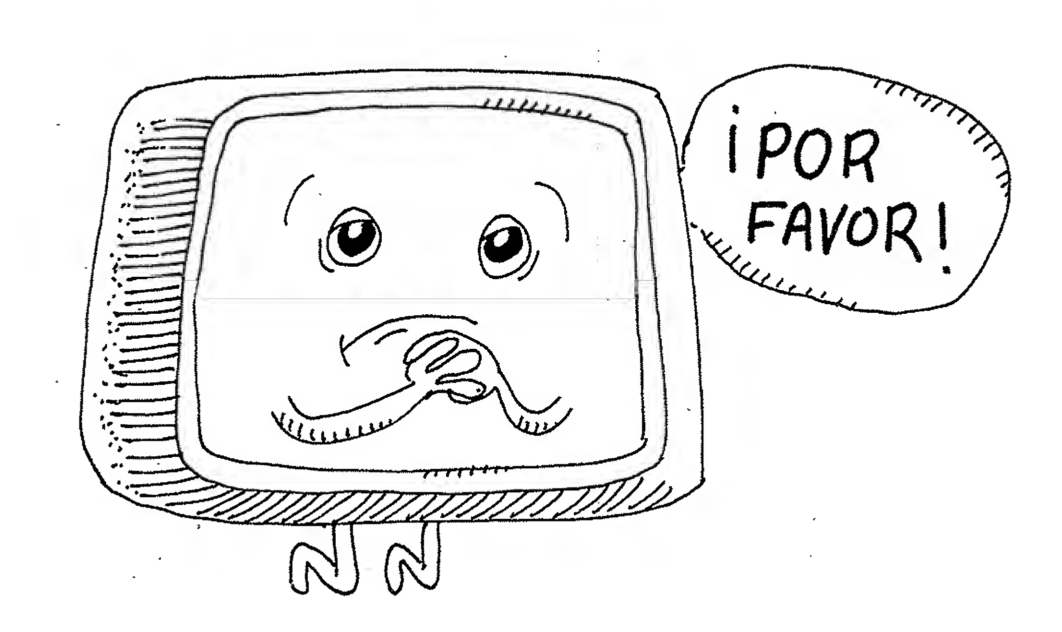
LA LECCION

Si, por el contrario, todo va bien y ya tienes delante:





A partir de ahora, por favor, sigue al pie de la letra mis instrucciones, las de la pantalla.



. . . de lo contrario te verás en crecientes dificultades.



En cualquier caso, ANIMO y ADELANTE. Esto es el principio. (No olvides que debes hacer una lección cada día, no intentes hacer más de una al día. . . podrías perder el tiempo).

Lección 2

Esta lección es un repaso de la lección 1 audiovisual. Basta con que la leas.

La lección 1 audiovisual es un presentación del ordenador. En ella sólo se pretende que empieces a perderle el miedo a una máquina que no muerde: el ordenador. El ordenador quiere ser tu amigo.

AVANCE – REPASO

1.– BLOAD "ATV", R

Esta es la instrucción que utilizarás todos los días para CARGAR la lección audiovisual.

La OPERACION DE CARGA consiste en traspasar los programas que hay almacenados en la caset, a la memoria del ordenador.

LOAD es una instrucción. Significa cargar la memoria del ordenador.

La MEMORIA del ordenador es una pieza en la que se almacenan datos.

2.– RETURN *Comando ejecutivo*

Es un COMANDO. Se pulsa después de haber tecleado alguna instrucción. RETURN es como decirle al ordenador: "¡Haz lo que te digo. Obedece!". Es una palabra ejecutiva.

3.– COMANDO

Un comando es una palabra que contiene una orden. Es una palabra mediante la cual ordenamos algo al ordenador. Ejemplos de comandos:

LIST, PRINT, BLOAD, LOAD

4.– TECLADO

Es la puerta de entrada al ordenador.

Por medio del teclado nos comunicamos con el ordenador. Tu MSX tiene seis teclados, y algunas teclas especiales que luego veremos.

5.– ESCRITURA

Nos comunicamos con el ordenador por medio del teclado. Cada tecla que pulsamos es entendida por el ordenador como una orden nuestra. Si tecleamos mal, el ordenador no entenderá y nos avisará del error, o no nos hará caso.

6.– SYNTAX ERROR

Es un mensaje con el que el ordenador nos avisa que nos hemos equivocado tecleando. El ordenador considerará ERROR todo aquello que no entienda.

7.– BS TECLA CORRECTORA

Sirve para corregir en el renglón que estemos escribiendo.

8.– O – Ø. 1-1

LETRAS

O	O	11
O	O	11

NUMEROS

Ø	Ø	11
Ø	Ø	11

9.– MAYUSCULAS

Tu MSX puede escribir en minúsculas y en mayúsculas. Esto último lo conseguiremos con las teclas de SHIFT. (En tu ordenador pueden aparecer como ↑).

También puedes conseguir letras en mayúsculas con la tecla sujeta-mayúsculas, que puede aparecer en tu ordenador así (↑), o así CAPS, e incluso tu ordenador te puede avisar de que trabaja en CAPS mediante una lucecita.

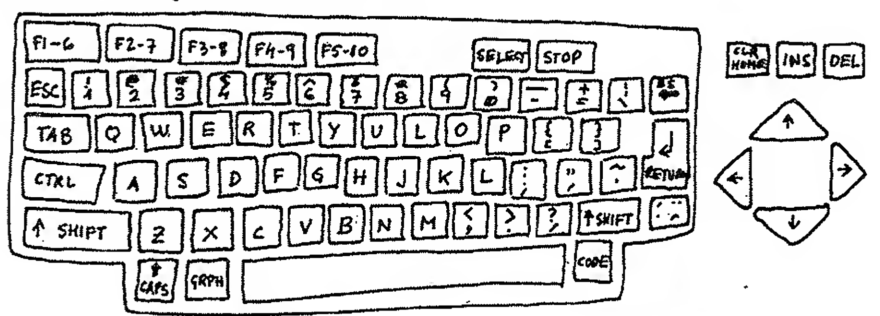
10.– GRPH *Gráficos*

Es la tecla de gráficos de tu ordenador. Con ella, tanto en minúsculas como en mayúsculas, puedes situar en la pantalla una serie de caracteres gráficos, que enriquecerán tus textos y programas facilitando tu trabajo.

TEMA DE HOY: EL TECLADO

¿Tienes delante el ordenador? ¿Está encendido y conectado a la pantalla?

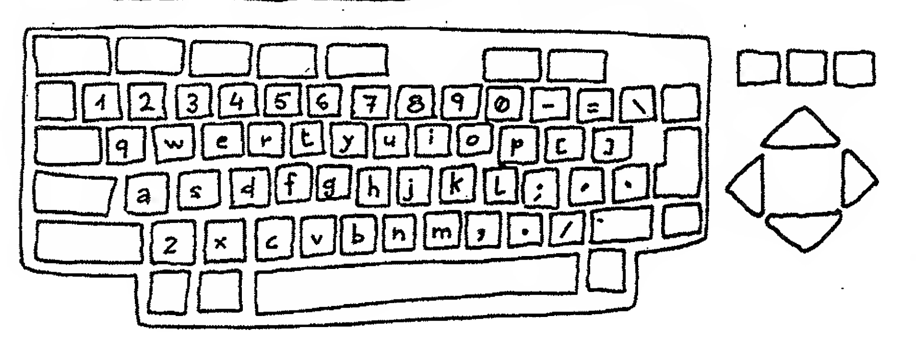
El teclado sirve para comunicarnos con el ordenador y hay que conocerlo bien. Hay que saber qué se oculta debajo de cada tecla.



TECLADOS DE TU MSX

El teclado de tu MSX es parecido al de una máquina de escribir, pero sólo parecido. Tu MSX dispone de seis teclados, que vamos a ver a continuación:

TECLADO TIPOGRAFICO EN MINUSCULAS

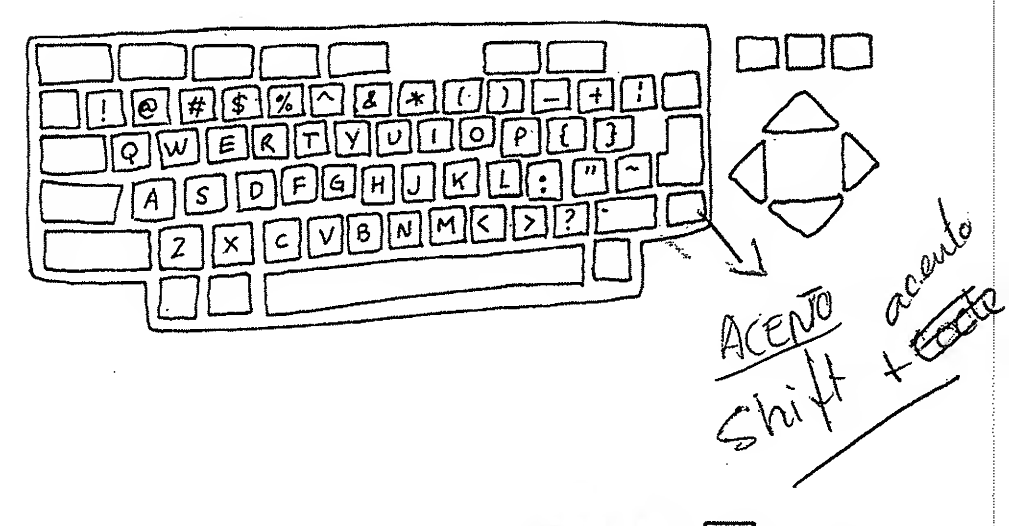


Pulsa una a una todas las teclas. Pulsa alguna y no levantes el dedo. ¡Sorpresa!

Pulsa, al mismo tiempo, diez teclas, y no levantes los dedos: ¡Sorpresa!

TECLADO TIPOGRAFICO EN MAYUSCULAS

Para obtenerlo basta que pulses la tecla de SHIFT ó [↑], y al mismo tiempo la tecla que desees. Con SHIFT se pueden escribir en mayúsculas las letras, o se pueden escribir los signos de la parte de arriba de las teclas que presentan dos signos:



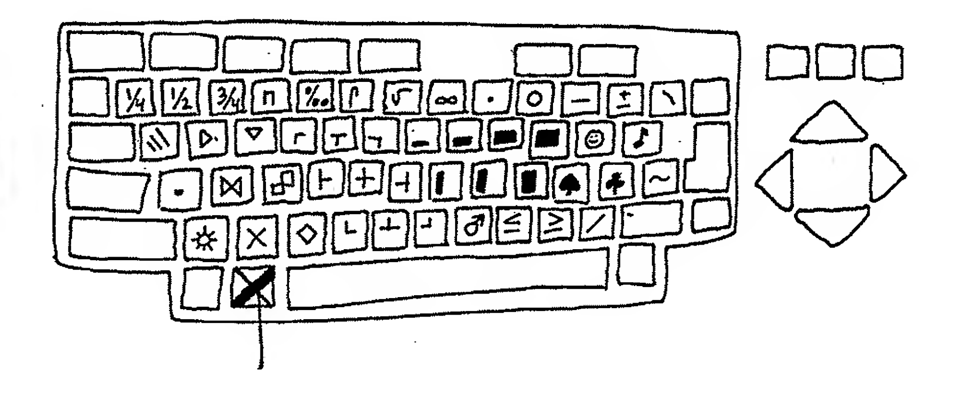
Con la tecla fijamayúsculas (CAPS ó [↑]) puedes escribir en mayúsculas todas las letras; sólo las letras.

El teclado tipográfico de mayúsculas tiene una particularidad interesante para nosotros: contiene el acento. Pulsando SHIFT y la tecla de acento y luego la vocal deseada, ésta saldrá acentuada.

Haz la prueba.

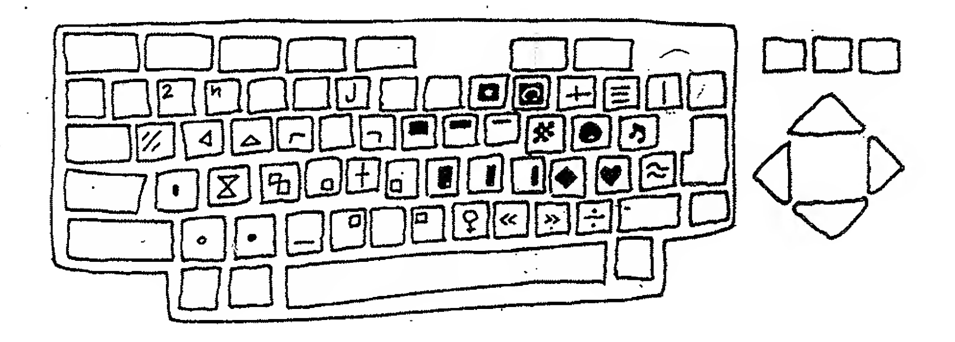
TECLADO DE GRAFICOS I

Se obtiene pulsando la tecla de GRPH que es la tecla de GRAFICOS, y simultáneamente, otra tecla cualquiera. Haz la prueba, y verás que te sale por la pantalla. (Si tienes la pantalla muy sucia, apaga y enciende el ordenador).



TECLADO DE GRAFICOS II

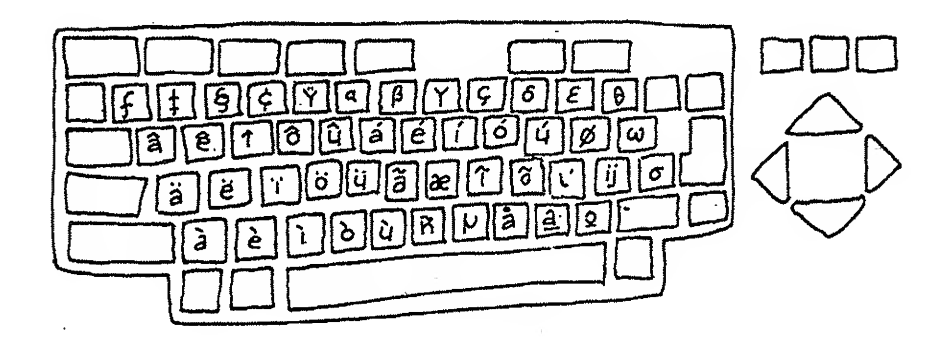
Lo obtendrás pulsando la tecla de GRPH y la de SHIFT ó ↑ Haz pruebas a ver qué nos oculta ese teclado.



TECLADO DE CARACTERES ESPECIALES I

Este teclado contiene letras griegas muy utilizadas en matemáticas, y algunos otros signos de interés. También contiene la ñ minúscula.



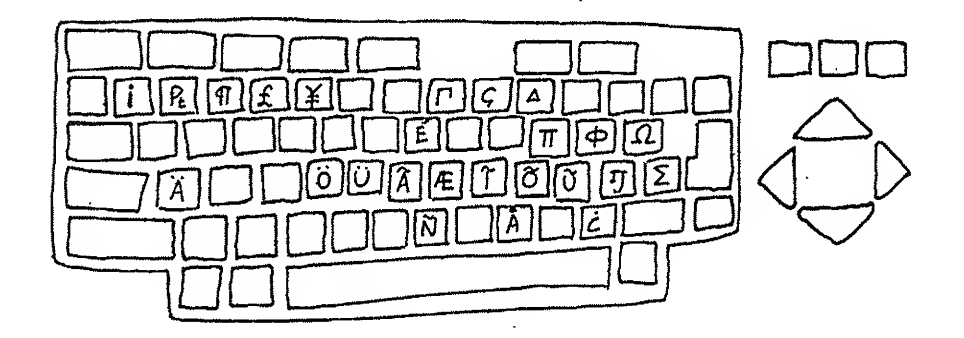


Sólo haciendo pruebas llegarás a conocerlo.

¡ADELANTE!

TECLADO DE CARACTERES ESPECIALES II

Entre otros signos contiene la EÑE mayúscula y la interrogación y admiración iniciales ¿, ¡. Se activa este teclado pulsando CODE y SHIFT o ↑.



Al final de la lección de hoy encontraremos algunos entretenimientos, que te ayudarán a conocer mejor el teclado.

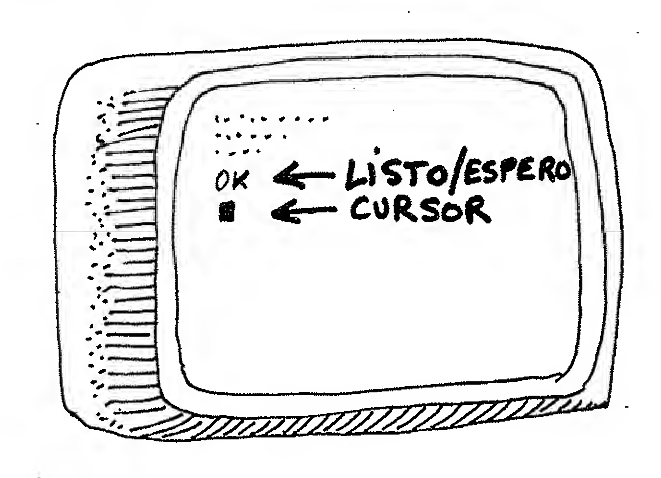


Si te equivocas corrige con BS que es la tecla correctora. (BACK SPACE).

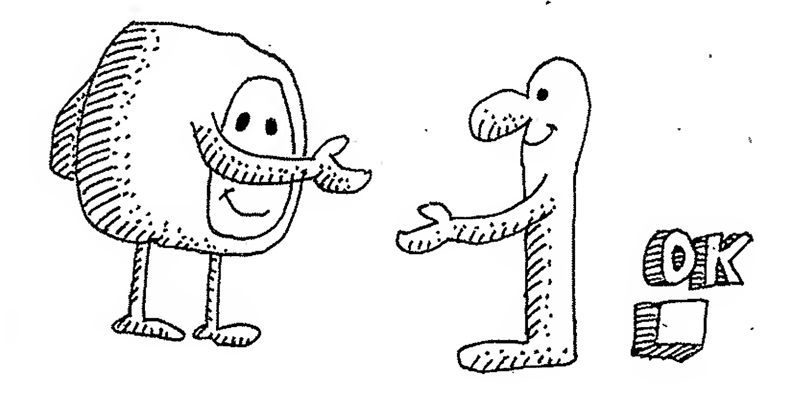
LA PANTALLA

Para comunicarte con el ordenador utilizas el T E C L A D O Pues bien, el ordenador se comunica contigo por medio de la pantalla.

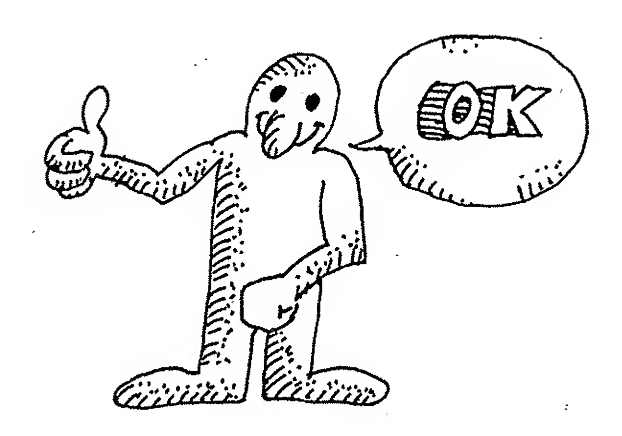
Enciende y apaga tu ordenador.



Tenemos que conocer la pantalla.



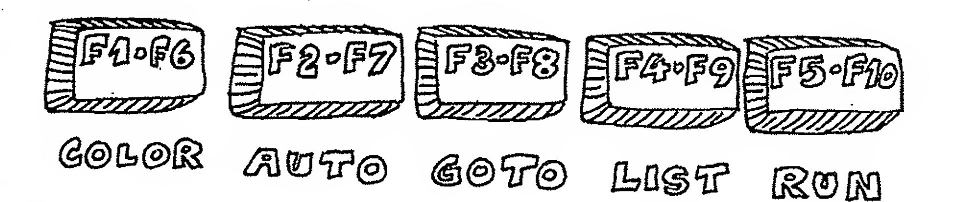
Vamos por partes. Siempre que en la pantalla aparece OK quiere decir que el ordenador está preparado para actuar. OK es el vigilante que nos indica que todo va bien.



Cuando en la pantalla aparece OK quiere decir que el ordenador ha terminado un proceso y que está listo para iniciar otro.

TECLAS DE FUNCION

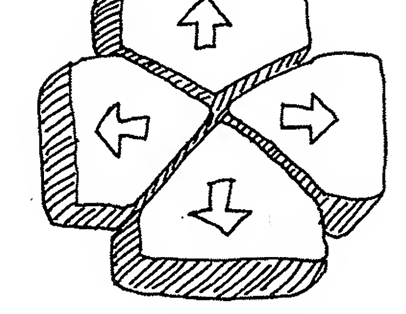
Como su nombre indican, son unas teclas que desempeñan una función concreta. Hoy me conformo con que te fijes, y sepas cuáles son:



Esas palabras son comandos de Basic, mandatos, órdenes. Cada una de las teclas de función contiene un mandato en minúsculas, y pulsando SHIFT otro mandato.

TECLAS DE CURSOR

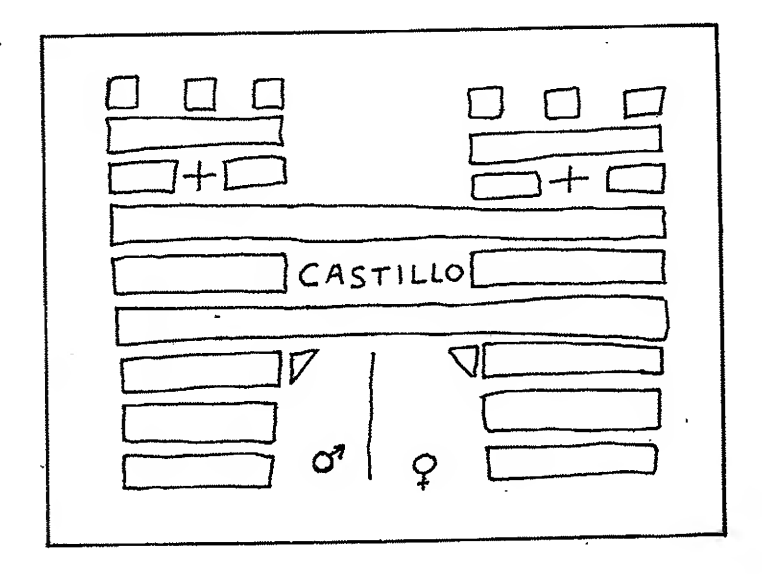
A la derecha de tu teclado hay cuatro teclas iguales.

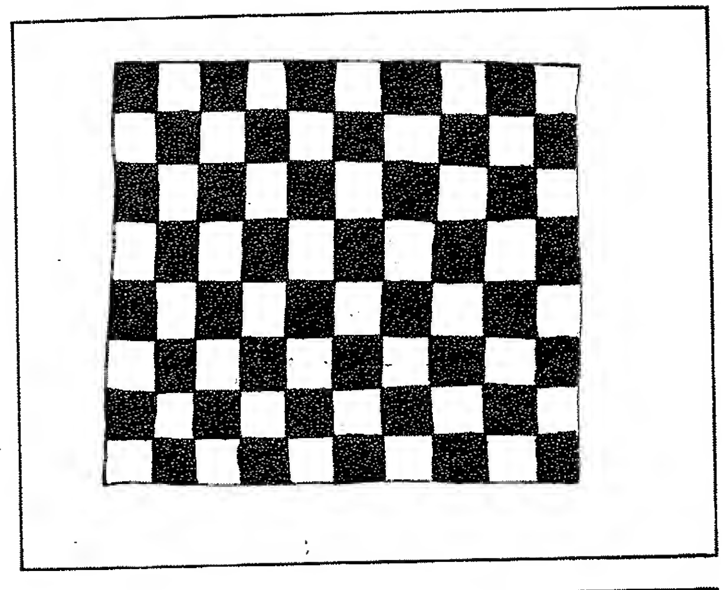


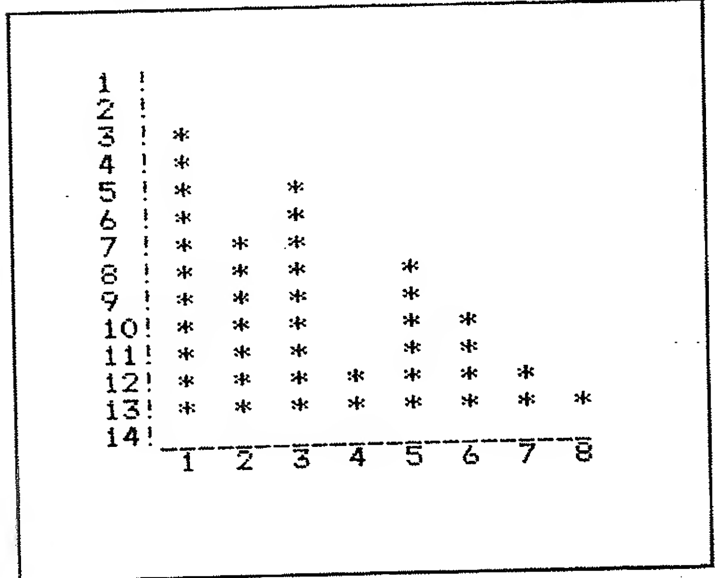
Son las teclas de Cursor. Con ellas puedes desplazar el cursor por la pantalla. Apaga y enciende tu ordenador con ON/OFF, y ahora desplaza el cursor por la pantalla en todas direcciones. Son las teclas de cursor. En lecciones próximas ampliaremos el conocimiento del teclado.

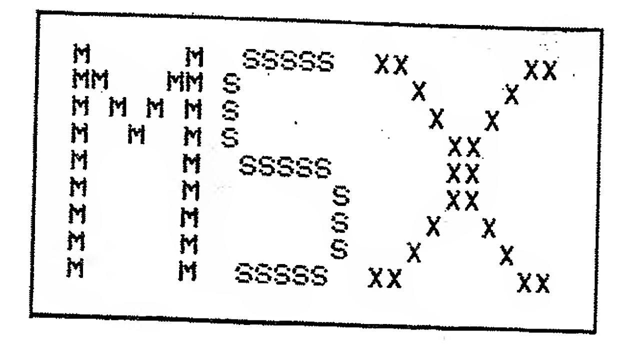
PRACTICAS TAREAS PROGRAMAS

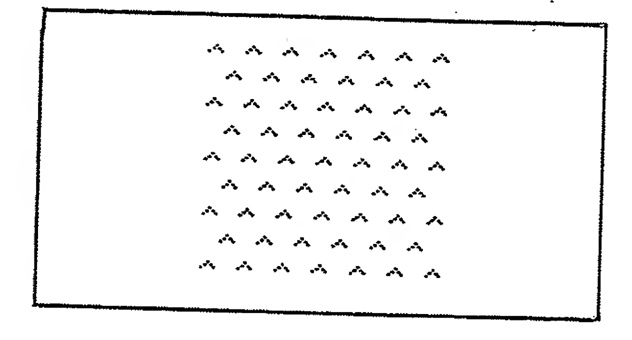
Utilizando el teclado y las teclas de cursor, a ver si puedes hacer en tu pantalla los siguientes dibujos.

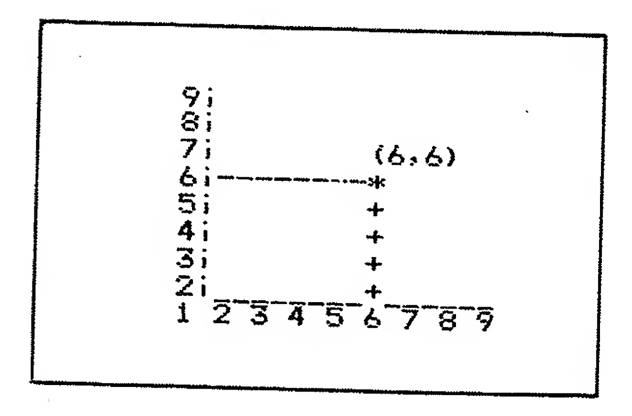












Lección 4

En las lecciones anteriores te hemos presentado el teclado; te hemos hablado del ordenador, del Basic. . . Hoy te vamos a presentar algunas teclas interesantes, de las que ya te hablamos en la lección 2ª Vamos a hablar también de rectificaciones y de programas.

AVANCE REPASO

1.– ORDENADOR

Un ordenador es una máquina que procesa datos: recibe datos y emite información. Las partes fundamentales del ordenador son la memoria y la unidad central de procesos.

2.– PROGRAMA

Conjunto de instrucciones y datos que se le introducen al ordenador por medio del teclado, para que resuelva alguna tarea.

3.– BASIC

Un lenguaje de programación para comunicarse con el ordenador.

4.– LINEA

Es un número, seguido de una instrucción, seguido de un dato. Los programas están formados por líneas.

2.– PROGRAMA

El ordenador es una máquina que procesa datos. Los datos se le suministran en forma de programas, que están compuestos por líneas, que contienen instrucciones y datos.

```
1Ø  A = Ø
2Ø  A = A + 1
3Ø  PRINT A
4Ø  GOTO 2Ø
```

Eso es un programa. Como verás, está formado por cuatro líneas: la 1Ø, la 2Ø, la 3Ø, la 4Ø. El número que se pone al principio de cada línea (1Ø, 2Ø, 3Ø, 4Ø) es el número de orden en que el ordenador irá leyendo las líneas del programa. Empezará por la línea de número más bajo, y terminará por la línea de número más alto: de menor a mayor.



Las líneas se numeran de 1Ø en 1Ø, por si se nos olvida algo, y después de terminar un programa, tenemos que añadirle más líneas.

Numerándolas de 1Ø en 1Ø, siempre tendremos la opción de introducir líneas intermedias: la 11, la 12, la 15, la 21, la 27, la 33, etc.

Bien. Un programa está formado por

------- -------- ---------------- -------- ---------

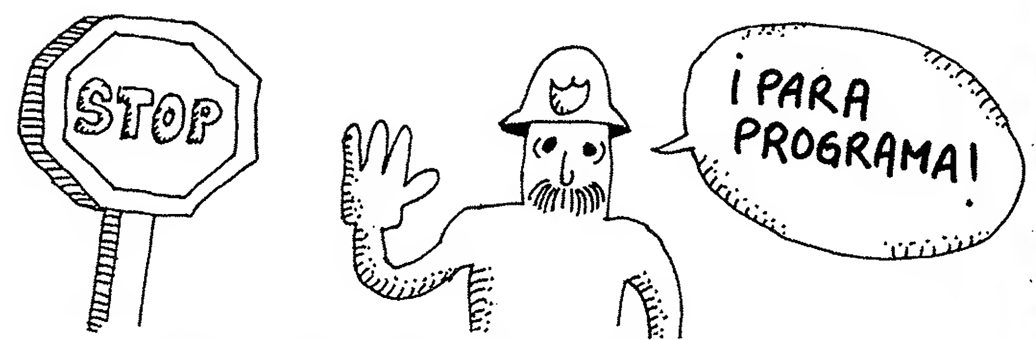
Sin perder tiempo, vas a copiar un programa: el que tienes delante. Apaga y enciende tu ordenador, y copia. Recuerda que después de terminar cada línea, debes pulsar la tecla RETURN.

Si has terminado ya, escribe en tu teclado RUN, y luego pulsa, RETURN. Si lo has hecho todo como te lo he dicho, en este momento tu ordenador está contando, y eso es lo que ves por la pantalla. La instrucción RUN significa "correr", y cuando se teclea RUN es como si se le dice al ordenador: "Adelante con el programa". Cuando tecleas RUN el ordenador ejecuta el primer programa que se encuentra en su memoria.

¿Sigue contando tu ordenador?.(Si no es así, copia nuevamente el programa, y ejecútalo).

Si el ordenador sigue contando, pulsa una tecla, que está en la fila superior del teclado, que dice STOP.

El ordenador ha dejado de contar. Ha parado.



Ahora, para que siga contando, vuelve a pulsar STOP

STOP detiene la ejecución de un programa, y si se vuelve a pulsar, lo reanuda.

Ahora, el programa sigue contando. ¿Ves a tu izquierda una tecla que dice CTRL?Púlsala, y sin dejar de pulsarla, pulsa STOP. El programa se ha detenido otra vez, y en la pantalla te aparece

BREAK IN 30.

STOP es como un guardia de tráfico, que detiene el tráfico un momento, y luego te permite seguir desde el número en que estabas.

CONTROL STOP, en cambio, es un guardia que te pone una multa, y no te deja continuar, hasta que le des una de dos contraseñas:

1. Si tecleas CONT y pulsas RETURN, continuarás el programa desde donde estabas.
2. Si tecleas RUN y tecleas RETURN empezarás desde el principio.

STOP
CTRL STOP

Teclea CONT y pulsa RETURN.

Acabamos de ver RUN, STOP, CTRL STOP, BREAK, CONT.

Pulsa CTRL STOP. Ahora pulsa la tecla de SHIFT o ↑ y otra que está arriba a la derecha en la que dice

C L R
H O M E

Acabas de limpiar la pantalla.



¿Habrá que copiar otra vez el programa?. . . No. Teclea LIST, y pulsa RETURN. ¡Ahí lo tienes otra vez! **LIST sirve para sacar por la pantalla el listado de un programa: es decir, sirve para sacar un programa por la pantalla.**

(RUN sirve para ejecutarlo, para rodarlo).

RUN pone en funcionamiento un PROGRAM-
A.

LIST saca por la PANTALLA el LIS-
TADO de un PROGRAMA.

CTRL STOP interrumpe la EJECUCIO-
N de un programa.

CONT sirve para CONTINUAR un pro-
grama.

STOP DETIENE la ejecución de un pro-
grama.

SHIFT CLR LIMPIA la PANTALLA.

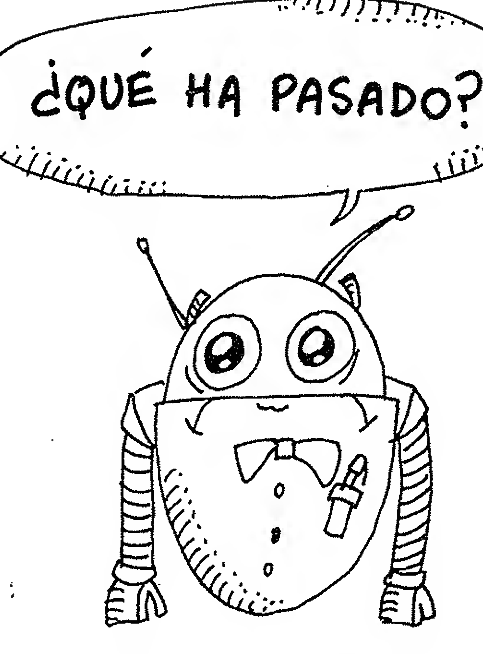
En estos momentos en tu pantalla tienes:

```
10  A = 0
20  A = A + 1
30  PRINT A
40  GOTO 20
```

Eso es un programa escrito en Basic. Pulsa SHIFT CLS: has vuelto a limpiar la pantalla. Ahora no ves el programa pero todavía permanece en la memoria del ordenador. Teclea LIST y pulsa RETURN.

Ahora, atento, teclea NEW, y pulsa RETURN. Ahora teclea RUN y pulsa RETURN. Teclea LIST y pulsa RETURN.

Ha pasado que **con NEW has borrado la memoria del ordenador.**
En este momento la memoria del ordenador está vacía.
Se le ha olvidado el programa que hay en pantalla.



Ahora:

SHIFT CLS

Ahora, copia literalmente el siguiente programa:

```
10  PRINT "ESTO ES"         RETURN
20  PRIMT "UN EJEMPLO"  RETURN
```

Teclea RUN pulsa RETURN.

SYNTAX ERROR IN 20

Vamos a corregir. Teclea LIST. RETURN. Utilizando las TECLAS DE CURSOR, sitúa el cursor sobre la línea 20.

20 PRIMT "un ejemplo"

El error sintáctico de esa línea, lo que está mal, es la M de PRIMT, que no es una M sino una N.

1º Lleva el cursor sobre la **M**.
2º Pulsa una vez la tecla **DEL** que está arriba a la derecha.
3º Pulsa la tecla **INS** que está al lado de **DEL**.
4º Pulsa la letra **N** en mayúscula.
5º Pulsa **RETURN**
6º Coloca el cursor por debajo de **OK**.

Has terminado la corrección. Ejecuta el programa para ver si funciona.

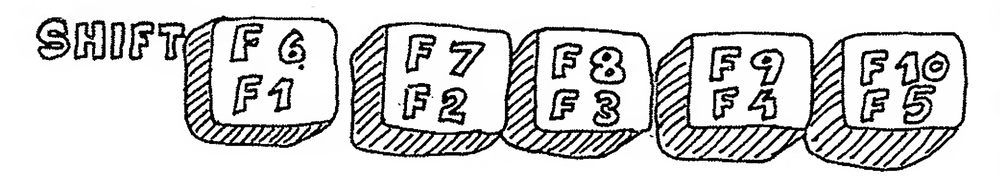
DEL **Borra errores de los programas.**

INS **Inserta caracteres donde le digamos.**

La instrucción con la que borramos la memoria del ordenador es R U N. Con I N S insertamos caracteres: en los programas, y con D E L borramos caracteres.

Cuando se acaba cada corrección se pulsa la tecla R E T U R N y antes de ejecutar el programa, hay que poner el cursor por debajo de la última línea del programa.

¿Recuerdas cuáles son las TECLAS DE FUNCION?



Teclea NEW. **RETURN** y **SHIFT** **CLS**

Copia ese programa (No olvides pulsar **RETURN** al final de cada línea).

```
1   COLOR 13
2   PLAY "L2C"
3   A$ = "MI NOMBRE ES EL TUYO"
4   PRINT A$
5   GOTO 1
```

Pulsa la tecla **F-5**. **F-5** es lo mismo que RUN + **RETURN**

Pulsa **CTRL** **STOP**.

Pulsa la tecla **F4** y luego **RETURN**

F4 **es lo mismo que LIST.**

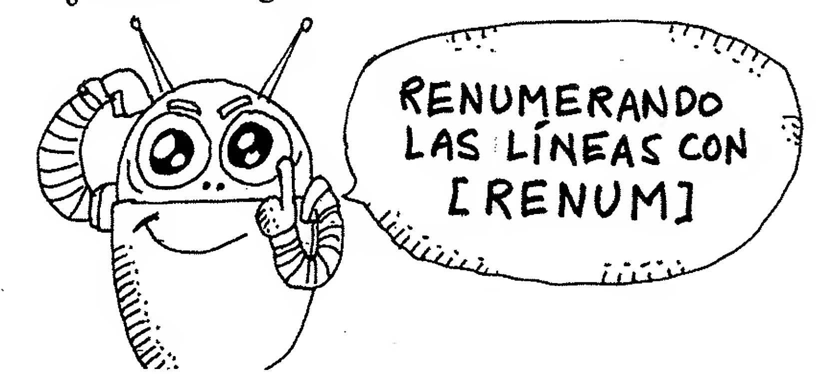
Pulsa **F5** Pulsa **CTRL** **STOP**.

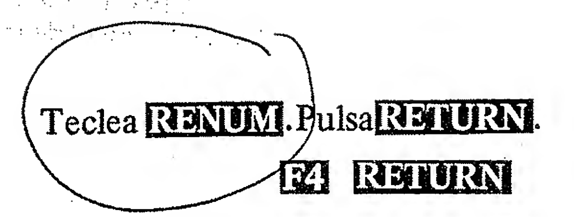
Pulsa ahora **SHIFT** **F8**.

SHIFT **F8** **es lo mismo que CONT** **RETURN**

Pulsa **F4** **RETURN** Si te fijas, las líneas del programa están numeradas de 1 en 1: 1, 2, 3, 4 y 5.

Yo quiero introducir una línea nueva entre la 2 y la 3. ¿Cómo lo hago?





Teclea RENUM. Pulsa RETURN.

F4 RETURN

¿Qué te parece? Ya tenemos las líneas numeradas de 1Ø en 1Ø.

Ahora vamos a introducir esta línea:

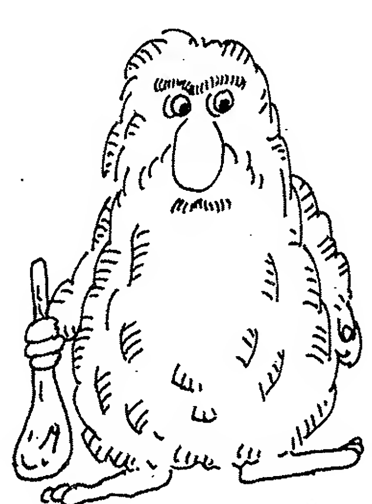
15 PLAY "L1 N1 N3Ø" RETURN.

Cópiala

F-5

Y ahora, en la línea 3Ø sustituye "Mi nombre es el tuyo", por tu nombre y apellidos (recuerda que debes ponerlos entre comillas).

Y ahora, un ejercicio salvaje: ¡Haz un resumen de esta lección!



RESUMEN PERSONAL

Pulsa F4. RETURN
Vamos a borrar la línea 5Ø. ¿Como?
Teclea 5Ø. RETURN ¡Ya está!

Tecleando el número de una línea, y pulsando luego RETURN se borra la línea en cuestión.

```
10 REM Distorsión
20 SCREEN 1
30 COLOR 0,15,1
40 RUN
```
1

```
1 REM " CON ESTE PROGRAMA TIENES QUE
JUGAR CONTRA EL ORDENADOR PARA VER QU
IEN TECLEA MAS RAPIDO"
10 TIME=0:A=RND(1)*224+31:A$=""
20 PRINT CHR$(A);" ";
30 IF TIME/50<5 THEN 40 ELSE 60
40 A$=INKEY$
50 IF A$="" THEN 30
60 IF CHR$(A)=A$ THEN PRINT A$:GOTO 1
70 PRINT A$:PRINT "FALLASTE"
80 GOTO 10
```
2

```
10 REM COLOR DEL 0 AL 15
20 SCREEN1
30 INPUT "DIME LAS CLAVES DEL COLOR";
N,N1,N2
40 COLOR N,N1,N2
50 PRINT "ESTAMOS VIENDO LA PANTALLA
EN"
60 PRINT "COLOR ";N;N1;N2
70 END
```
3

Con la instrucción COLOR se puede cambiar el color de pantalla, y de las letras. Tu **MSX** dispone de 16 colores numerados del Ø al 15. Esta instrucción la tenemos también en las teclas de función:

F1 = COLOR

FUNCIONES DE LA TECLA CTRL

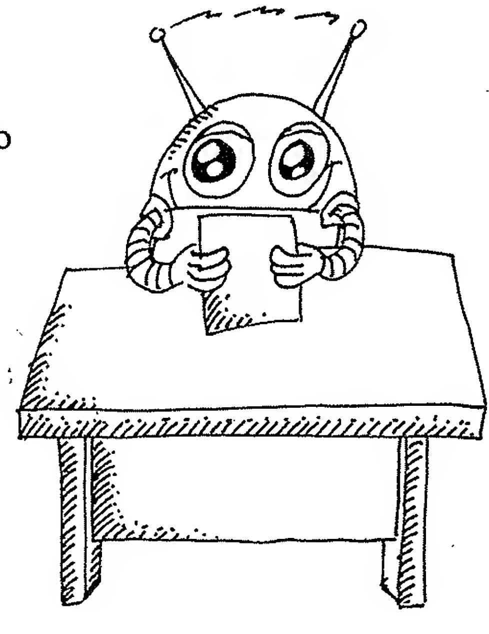
Además de las teclas de edición, el BASIC **MSX** ofrece funciones especiales, simplemente pulsando la tecla **CTRL** simultáneamente con otra.

Teclas Pulsadas	Función
CTRL + B	Mueven el cursor hasta el comienzo de de una palabra (grupo de caracteres separados por un espacio). Cuando el cursor esté al comienzo de una palabra, se moverá hasta el comienzo de la inmediatamente anterior.
CTRL + C	Desactivan el estado en espera de introducción, o la generación automática de números de línea activada mediante el mandato AUTO, para volver al estado de espera de mandatos.
CTRL + E	Borran desde la posición del cursor hasta la última línea del programa.
CTRL + F	Mueven el cursor hasta el comienzo de la palabra siguiente.
CTRL + G	Generan un pitido.
CTRL + H	Igual que la tecla BS.

CTRL + I	Igual que la tecla TAB.
CTRL + J	Mueven el cursor 1 línea hacia abajo.
CTRL + K	Igual que HOME.
CTRL + L	Igual que SHIFT + HOME
CTRL + M	Igual que la tecla RETURN.
CTRL + N	Mueven el cursor hasta el punto siguiente al último carácter de una línea.
CTRL + R	Igual que la tecla INS.
CTRL + U	Borran todos los caracteres de una línea.
CTRL + X	Igual que SELECT. Sin aplicación en el BASIC MSX
CTRL + \	Igual que ◁
CTRL + [	Igual que ESC Sin aplicación en el BASIC MSX
CTRL +]	Igual que ▷
CTRL + Λ	Igual que ◁
CTRL + –	Igual que △

Lección 6

Hoy vamos a darnos un paseo por el interior del ordenador. Vamos a curiosear sus piezas para comprender un poco el funcionamiento de ese montón de chips, circuitos integrados, circuitos impresos y tornillos.
Y vamos a ver también cómo funciona la memoria, y cómo se guardan los datos en su interior.



Pero antes de nada, veamos cómo aumenta nuestro acervo informático.

AVANCES REPASO

1– RUN. Es la instrucción con la que echamos a andar un programa. Sirve para rodar un programa, para que corra, para que se ejecute.

2– STOP. Es una tecla con la que detenemos y reanudamos un programa que está funcionando.

3– CTRL STOP. Produce un BREAK (una ruptura) en el programa.

4– CONT. Reanuda un programa interrumpido con CTRL STOP, desde donde se quedó interrumpido.

5– TECLAS DE FUNCION. De la F-1 a la F-10, son teclas que contienen una instrucción.

6– LIST. Es una instrucción o comando con la que obtenemos en pantalla un listado de los programas que hay en la memoria del ordenador.

7– SHIFT CLR. Limpia la pantalla.

8– NEW. Limpia la memoria del ordenador (ON/OFF también).

9– DEL Borra caracteres de una línea de programa. BS. Borra caracteres cuando aún no se ha cerrado una línea de programa.

10– INS. Sirve para insertar caracteres en una línea de programa.

11– COLOR. Es una instrucción que sirve para elegir el color de la pantalla.

12– PROGRAMA. Es un conjunto de instrucciones y datos, que se dan al ordenador para que los ejecute. Los programas están formados por líneas.

13– LINEA. Está formada por el núm. de línea, instrucción y dato.

14– NUMERO DE LINEA. Se pone en cada línea para indicar al ordenador en qué orden debe leerlas. Se suelen numerar las líneas de 1Ø en 1Ø.

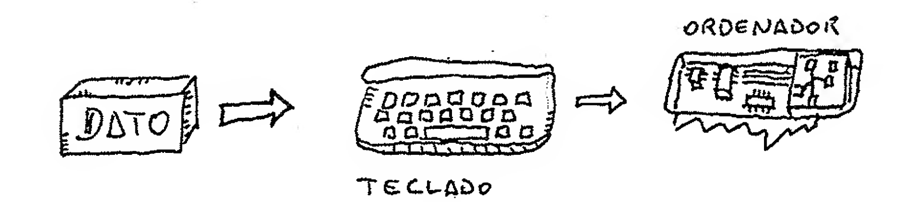
15– RENUM. Es una instrucción que sirve para renumerar las líneas de un programa.

VAMOS AL GRANO

La informática es la ciencia que trata datos mediante ordenador; es la INFORmación autoMATICA. La información tratada con máquinas, el proceso de datos.

Un ordenador es una máquina que procesa datos.

Los datos entran al ordenador por medio del teclado, en forma de letras, números, caracteres.

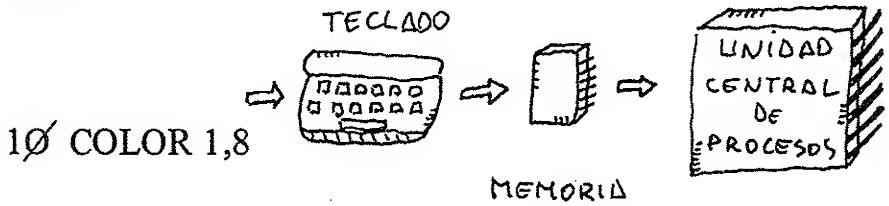


Imaginate que con la instrucción COLOR y el dato 5, queremos cambiar el color de la pantalla. Teclea 1Ø COLOR 1, 8 RETURN Le hemos dado al ordenador una instrucción y unos datos por el teclado. Del teclado,

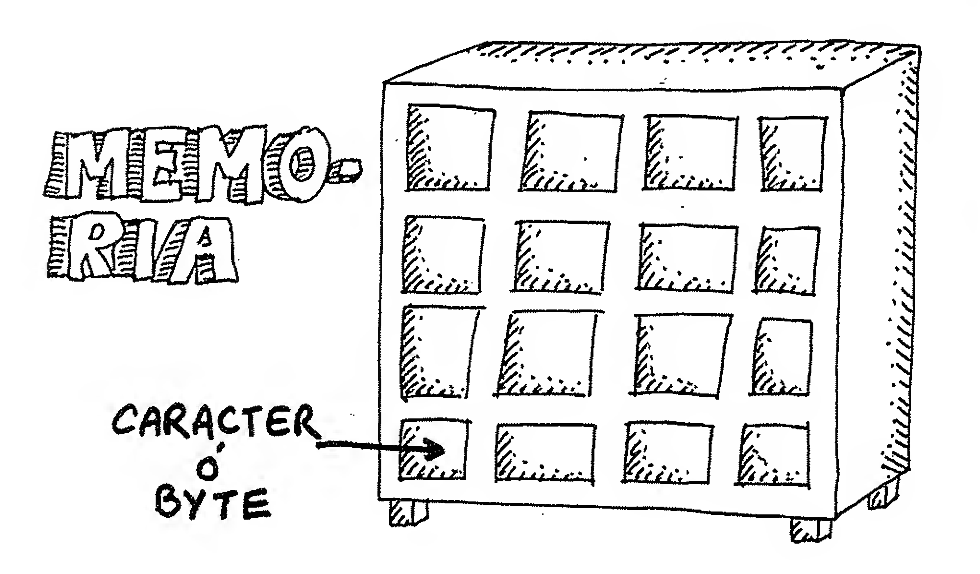
al pulsar RETURN, esa línea 1Ø ha pasado a la memoria de tu ordenador, y allí se ha quedado una copia, y la prueba es que si borras la pantalla, y tecleas LIST, la línea vuelve a aparecer. Adelante.

Ahora, pulsa F-5. Ahora para, volver al color original, pulsa SHIFT F-6

Cualquier orden que tú das al ordenador sigue este camino:



La MEMORIA es como un gran archivo, en el que se guarda todo lo que tú tecleas.

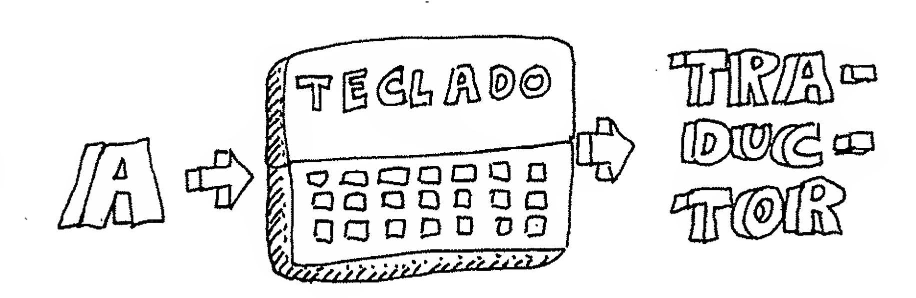


En cada uno de esos cajones de la memoria se guarda un carácter o BYTE. En un BYTE cabe una letra o un número, una instrucción etc.

La memoria está formada por BYTES. Cada BYTE, a su vez, está formado por 8 BITS.

¿Qué es un BIT? Un BIT es la 1/8 parte de un BYTE. Con un ejemplo lo comprenderás. La letra A por ejemplo, ocupa un BYTE en la memoria. Tú tecleas en tu ordenador A y la letra, va desde el teclado hacia la memoria.

En el vestíbulo de la memoria hay un individuo informático que se llama TRADUCTOR. El traductor habla en idioma BASIC, y toma la letra A y la traduce al



idioma de los ordenadores, que es el lenguaje de máquina. El LENGUAJE DE MAQUINA es muy curioso,

porque en él cada letra, cada carácter está formado por un conjunto de unos y ceros. Por ejemplo: la letra [A] en código máquina sería:

Ø 1 Ø Ø Ø Ø Ø 1

Entonces, cuando tu tecleas A, el traductor la traduce a código máquina y le dice al ordenador:

Ø 1 Ø Ø Ø Ø Ø 1

Cada uno de esos números, aparece en la pantalla como un puntito. La [A] sería:

A = Ø 1 Ø Ø Ø Ø Ø 1 = ⁘

TU TECLEAS	EL TRADUCTOR TRADUCE	EL ORDENADOR ENTIENDE

Tú le hablas al ordenador en BASIC. El ordenador entiende en lenguaje de máquina.

El abecedario de los lenguajes europeos contiene 29 ó 3Ø letras. El abecedario del lenguaje de máquina contiene 256 caracteres. El abecedario del lenguaje de máquina no se denomina abecedario, sino CODIGO ASCII. Vamos a verlo. Copia y ejecuta este programa:


```
1Ø FOR C = Ø TO 255
2Ø PRINT CHR$ (C)
3Ø NEXT C
```

RUN RETURN



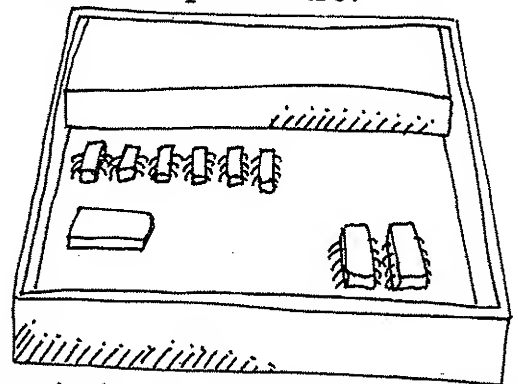
No te preocupes si te da la impresión de que te estamos hablando en chino en vez de en BASIC. Esto no es difícil, pero tienes que leerlo despacio. Te lo voy a repetir, a ver si ahora lo entiendes mejor.

Tú te comunicas con el ordenador en BASIC. Los ordenadores hablan en lenguaje de máquina. El traductor es una parte del ordenador que traduce tus instrucciones de Basic a lenguaje de máquina. Tú dices A, el traductor dice Ø 1 Ø Ø Ø Ø Ø 1, y el ordenador entiende ⁘ . Cada uno de esos puntitos es un BIT, **ocho bits forman un BYTE**, carácter u octeto.

Cada mil BYTES forman un KILOBYTE. (Realmente cada KILOBYTE tiene 1.024 BYTES, pero por comodidad consideraremos que solo tiene 1.000).

La capacidad de memoria de los ordenadores se mide en KILOBYTES, también llamados K'S.

Los BYTES son como casilleros o cajones de la memoria en los que guardamos las instrucciones, datos etc. Esto es un ordenador por dentro.



Como verás, lo que tiene por dentro son piezas. Una de esas piezas es la MEMORIA.

Bien. Borra la memoria de tu ordenador con NEW. Ya conoces los nombres de varias instrucciones.

C O L O R _ U _ _ _ W

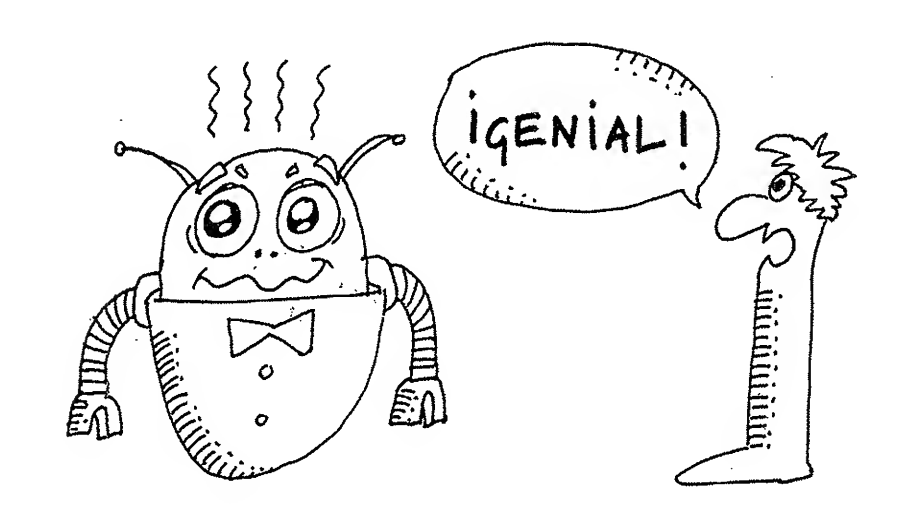
L I S T Y conoces muchas más palabras relacionadas con la informática. B I T, B Y T E M E M O R I A, K I L O B I T E.

El abecedario de los ordenadores se denomina código A S H I I y su lenguaje es lenguaje de M A Q U I N A. El abecedario del código ASCII contiene 2 5 6 caracteres.

DATOS

Te hemos presentado la memoria. Un ordenador es un procesador de datos. Es una máquina que trabaja con datos. Los datos pueden almacenarse de dos formas: como constantes, y como variables. Hablemos de ellos. Los datos se pueden almacenar en forma de C O N S T A N T E y en forma de V A R I A B L E.

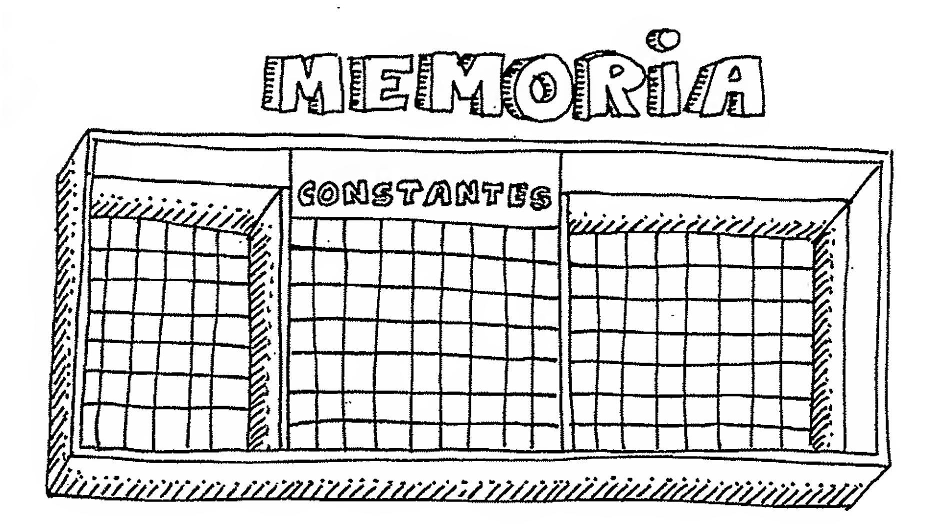
¿Qué quiere decir CONSTANTE?
Constante es algo que no varía.



Copia este ejemplo:

```
10   PRINT   5          RETURN
20   PRINT   "HOLA"     RETURN
F5
SHIFT   CLR
```

Ejecuta varias veces el programa con F5 . Cada vez que lo has ejecutado, PRINT, que es una instrucción que ya veremos, nos da el mismo valor: 5, y Hola. 5 y Hola son dos datos constantes: no varían. 5 es un dato NUMERICO, y HOLA es un dato LITERAL. Siempre que ejecutemos ese programa, aparecerán con el mismo valor. **Un dato constante es aquel que a lo largo de la ejecución de un cálculo, de una operación, de una tarea, de un programa, no varía.** Los datos constantes tienen una zona de la memoria para ser almacenados.

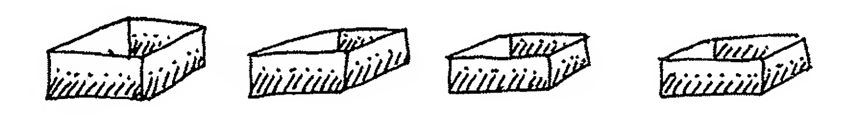


Todos son ejemplos de datos constantes: 5, 33, Hola, Mañana, 5 * 8 * 3 + 4 – 2, ☺, ■ , etc.

¿Y una variable qué es?

Es todo lo contrario. Del mismo modo que los datos constantes se almacenan en un lugar de la memoria, las VARIABLES son unos casilleros o cajones, especiales de la memoria, donde vamos a guardar datos.

Las variables no son datos. Son lugares de la memoria donde guardaremos datos.



Las variables son como pequeños depósitos donde echaremos datos. A cada variable le daremos un nombre, el que queramos (ver apéndice de esta lección), siempre que empiece por una letra: NUNCA PUEDE EMPEZAR EL NOMBRE DE UNA VARIABLE POR UN NUMERO.

Ejemplos de nombre de variable:

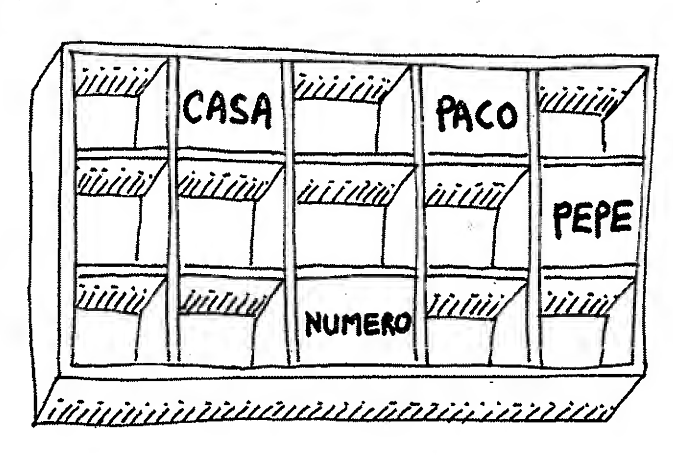
PACO, PEPE$, HR1, TG, X$, XX1, XX2%, CASA, RELOJ%, NUMERO 1.

Ahora ejecuta el siguiente programa:

```
10 PACO = 140
20 PEPE = 513
30 CASA = 329
40 NUMERO = PACO + PEPE + CASA
50 PRINT NUMERO
60 END
```

El resultado de ese programa es 982. En las líneas 1Ø, 2Ø, 3Ø y 4Ø has dado valor a las variables PACO, PEPE, CASA y NUMERO. (La línea 5Ø la veremos más adelante, aunque la curiosidad es la virtud más importante de un programador).

PACO, PEPE, CASA, NUMERO, son nombres de variables. En este momento, en la memoria de tu ordenador hay cuatro espacios reservados para PACO, PEPE, CASA, NUMERO.



El contenido de esos cuatro casilleros variará si tú cambias los números que has guardado en ellos, pero en la memoria de tu ordenador seguirá habiendo un espacio reservado para PACO, PEPE, CASA y NUMERO. Por eso las variables son variables: PORQUE SU CONTENIDO PUEDE VARIAR.

Veamos cómo vas tú de memoria:

V A R I A B L E, C O N S T A N T E

Un ordenador es una máquina que funciona con datos. Los datos se almacenan en la memoria, de dos modos: en forma de constantes, o en forma de variables. A las variables se les pone nombre, siempre que no empiecen por un número (ver apéndice de la lección).

Los datos pueden ser de dos tipos:

NUMERALES y todos los demás.

<u>Numerales</u>: son aquellos que sirven para hacer operaciones de cálculo.

<u>Todos los demás</u>: palabras, caracteres gráficos, números que no sirvan para hacer operaciones.

Se distinguen de un modo muy fácil:

NUMERALES: 5Ø, 8, 3 * 5 – 3, Ø.3 no comill

LOS DEMAS: “5Ø”, “HOLA”, “⎍”, “3 días”.

Los Numerales **no van** entre comillas. Los “demás” sí. A los “demás” los vamos a llamar datos “ALFANUMERICOS”, o “DE CADENA”.

A los numerales, para no cambiarles mucho el nombre, los vamos a llamar datos NUMERICOS.

Y para seguir inventando nombres, a las variables que guardan datos NUMERICOS las llamaremos VARIABLES NUMERICAS, y, a las variables que guardan datos “ALFANUMERICOS” o “DE CADENA” las llamaremos VARIABLES “ALFANUMERICAS” o “DE CADENA”.



PRACTICAS TAREAS PROGRAMAS

4

```
10 PRINT"*"
20 PRINT"**"
30 PRINT"***"
40 PRINT"****"
50 PRINT"*****"
60 PRINT"******"
70 PRINT"*******"
80 PRINT"********"
90 PRINT"*********"
100 END
```

5

```
10 PRINT"ESTO ES"
20 PRINT"UN EJEMPLO"
30 PRINT"DE COMO FUNCIONA"
40 PRINT"EL COMANDO PRINT CON COMILLA
S"
50 END
```

6

```
10 REM"PROGRAMA  "
20 SCREEN1
30 COLOR 12,7,11
40 PLAY"N13N14N16N16N16N14N13N13N13N1
3N14N16N16N16N14N13N13N13N14N16N18N20
N18N20N18N16N18N16N18N16N14N13N13"
50 END
```

7

```
10 REM"PROGRAMA  "
20 SCREEN1
30 COLOR 12,7,11
35 PRINT"CANCION DE RADIO"
40 PLAY"N18N16N15N18N15N16N18N20N18N1
6N15N13N16N15N13N15N16N18N15N13N15N18
N15N13N11"
50 END
```

8

```
10 REM"PROGRAMA  "
20 SCREEN1
30 COLOR 3,10,15
40 PLAY"N1N3N5N6N8N10N12N13"
50 PLAY"N13N12N10N8N6N5N3N1"
60 RUN
```

9

```
10 REM"PROGRAMA  "
20 SCREEN1
30 COLOR 7,4,2
40 PLAY"N8N20N12N15N13N9N16N29N69N69N
18N1"
50 LIST
```


10

```
10 REM MALLA CON LINE
20 SCREEN 2:COLOR 15,4,4:CLS
30 A=2:B=2:Y=10:X=10:X1=232:Y1=172
40 X=X+A:X1=X1+-A
50 Y=Y+B:Y1=Y1+-B
60 IF X<12 OR X>232 THEN A=-A
70 IF Y<12 OR Y>172 THEN B=-B
80 LINE(X,Y)-(X1,Y1),15
90 GOTO 40
```

```
10 REM EJEMPLO DE DRAW,ESCALA Y COLOR
20 SCREEN 2:COLOR 15,4,4:CLS
30 FOR S=1 TO 60
40 C=C+1:IF C>15 THEN C=1
50 DRAW"C"+STR$(C)+"S"+STR$(S)+"BM120
,120NL5BU10NR5L5D10R10U10"
60 NEXT S
70 GOTO 30
```

11

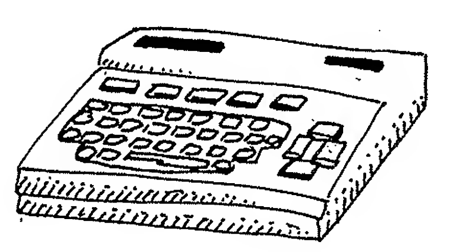
Apéndice. Palabras reservadas

Ninguna de estas palabras puede utilizarse como nombre de variable, porque daría SYNTAX ERROR.

ABS	DATA	IF	NAME	SAVE
AND	DEF	IMP	NEW	SCREEN
AS	DEFINT	INKEY$	NEXT	SGN
ASC	DEFDBL	INP	NOT	SIN
ATN	DEFSNG	INPUT	OCT$	SOUND
AUTO	DEFSTR	INPUT$	OFF	SPACE$
BASE	DEFUSR	INSTR	ON	SPC
BEEP	DELETE	INT	OPEN	SPRITE
BIN$	DIM	INTERVAL	OR	SPRITE$
BLOAD	DRAW	KEY	OUT	SQR
BSAVE	DSKF	KILL	PAD	STEP
CALL	ELSE	LEFT$	PAIN	STICK
CDBL	END	LEN	PDL	STOP
CHR$	EOF	LET	PEEK	STR$
CINT	EQV	LINE	PLAY	STRIG
CIRCLE	ERASE	LIST	POINT	STRING$
CLEAR	ERL	LLIST	POKE	SWAP
CLOAD	ERR	LOAD	POS	TAB
CLOSE	ERROR	LOC	PRINT	TAN
CLS	EXP	LOCATE	PSET	THEN
COLOR	FIELD	LOF	PRESET	TIME
CONT	FILES	LOG	PUT	TROFF
COPY	FIX	LOPS	READ	TRON
COS	FN	PRINT	REM	USING
CSAVE	FOR	LSET	RENUM	USR
CSNG	FRE	MAXFILES	RESTORE	VAL
CSRLIN	GET	MERGE	RESUME	VARPTR
CVD	GOSUR	MID$	RETURN	VDP
CVI	GOTO	MKD$	RIGHT	VPEEK
CVS	HEX$	MKI$	RND	VPOKE
		MOD	RSET	WAIT
		MOTOR	RUN	WIDTH
		MKS$		XOR

Lección 8

Hoy vamos a seguir hablando de las variables, y continuaremos el paseo por el interior del ordenador. También hablaremos de otras máquinas. Ten en cuenta algo: todas nuestras explicaciones van encaminadas a que comprendas que un ordenador es una máquina, una herramienta para ser utilizada, por cualquier persona. No hace falta saber matemáticas ni ser un genio. Solo hace falta tener curiosidad e interés por el aprendizaje.



AVANCE REPASO

1– BASIC. Es un lenguaje de programación para comunicarnos con el ordenador.

2– LENGUAJE MAQUINA. Es el idioma que habla el ordenador en su interior.

3– CODIGO ASCII. Es el abecedario del lenguaje máquina. Contiene 256 caracteres.

4– BIT. Es la unidad mínima de información. Cada 8 bits forman un byte o carácter.

5– BYTE. También llamado OCTETO (8 bits) es el lugar que ocupa en la memoria un carácter.

6– KILOBYTE. También llamado K de memoria. Contiene 1.000 bytes. La capacidad de memoria de los ordenadores se mide en Kilobytes.

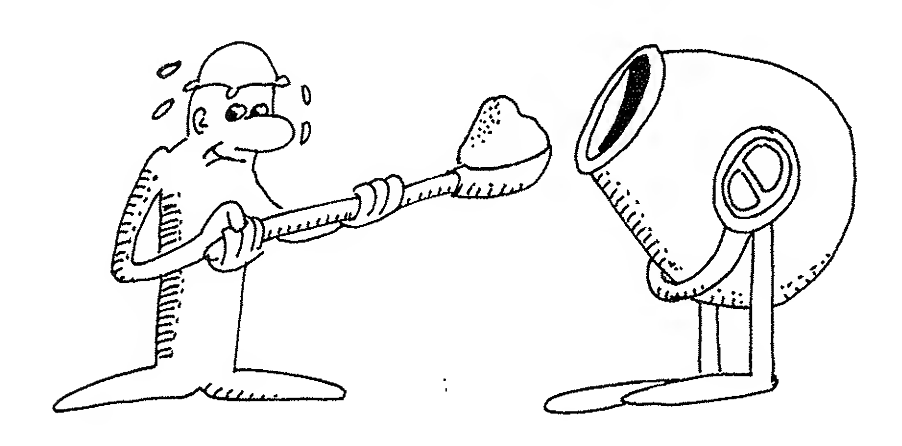
7– DATO. Son los números y letras por medio de las cuales suministramos al ordenador información. Pueden ser NUMERICOS (números para hacer operaciones) y ALFANUMERICOS (cadenas de caracteres) "entre comillas".

8– CONSTANTE. Es un tipo de dato. Es un dato que no varía a lo largo de todo el programa.

9– VARIABLE. Es un lugar de la memoria en el que podemos guardar un dato. A las variables se les pone nombre: A, B, PR, AB$, W, T123, etc. El nombre de una variable nunca empieza por un número, ni puede contener una palabra reservada.

Las variables pueden ser NUMERICAS y ALFANUMERICAS.

Y ahora presta mucha atención, porque vamos a entrar en faena. Vamos a suministrarte más datos.

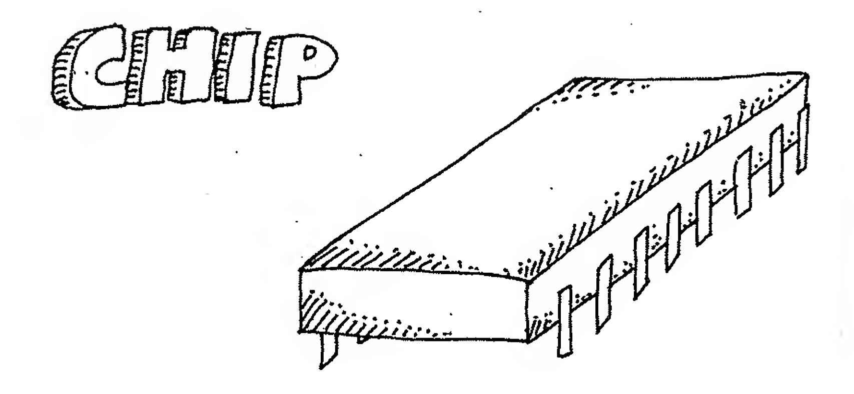


HOY VEREMOS QUE

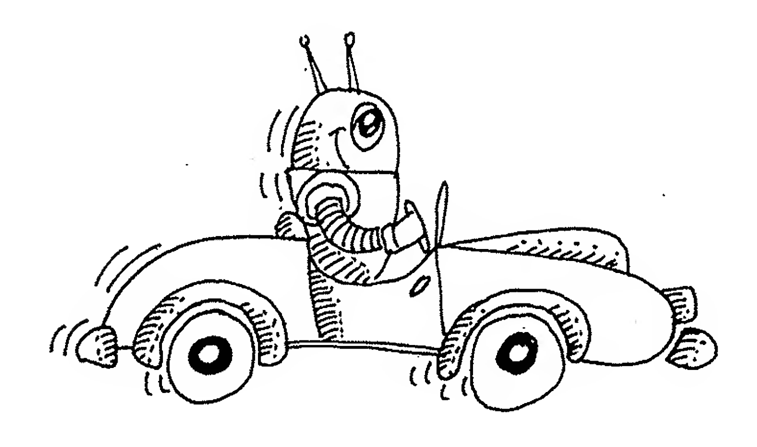
El BASIC es el lenguaje mediante el cual nos comunicamos con el ordenador. Es un idioma. El BASIC MSX es como un dialecto. El MSX es el BASIC más moderno, y el que sirve para trabajar con más ordenadores.

Bien, continuemos con la MEMORIA.

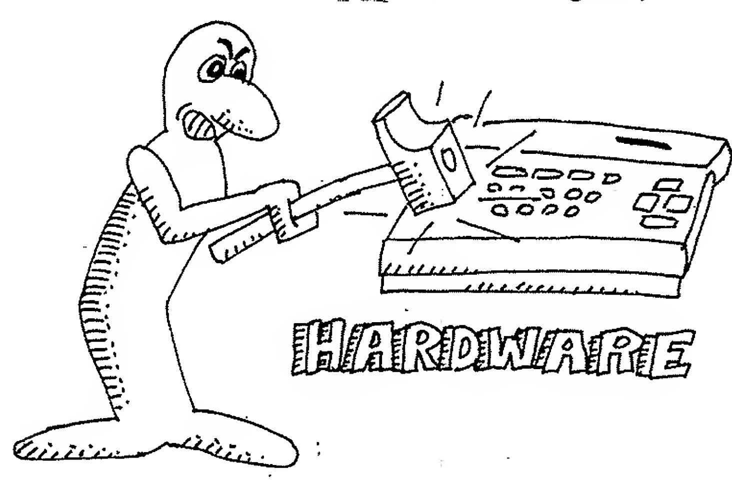
El ordenador funciona con datos. Los datos son el combustible de los ordenadores. Igual que un coche funciona con gasolina, un ordenador funciona con datos.



Los datos llegan a la memoria, y se guardan como constantes, o en variables. Pero vamos a ver: ¿La memoria tiene cajones de verdad? No. La memoria es una pieza, un CHIP.



Tu ordenador por dentro tiene piezas. A las piezas de tu ordenador, y al ordenador en sí se le llama genéricamente, HARDWARE (parte dura, o tangible).



Pero HARDWARE no son solo los ordenadores. También son HARDWARE los periféricos.

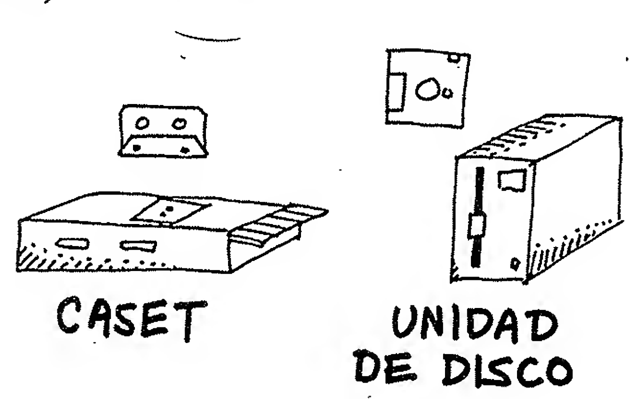


PERIFERICO es cualquier dispositivo que se conecta a un ordenador, para ayudarle en su trabajo. Un periférico es un auxiliar del ordenador.

El periférico que más conoces es la pantalla. También conoces el caset.

El caset es una memoria auxiliar de tu ordenador. La memoria de tu ordenador se llama memoria interna, porque está dentro de la carcasa del ordenador. El caset es memoria externa o auxiliar. Todo lo que no cabe en la memoria interna, lo puede guardar en la memoria externa.

Otra memoria externa es la Unidad de Disco. (En el manual de tu MSX encontrarás especificaciones sobre su uso).



El caset y la unidad de disco son memorias auxiliares de tu ordenador.

capacidad

Tu ordenador tiene dos memorias internas: La RAM y la ROM. En la ROM contiene las instrucciones que le han grabado en la fábrica del ordenador. Esa memoria siempre está llena, y es como el motor del ordenador.

La RAM es como el depósito de gasolina del ordenador, y ahí guardamos los datos. Cuando ya no caben más datos en la MEMORIA RAM, podemos guardarlos en el caset o la unidad de disco. (Ver apéndice de la lección).

Bien cambiemos de tema, pero hablando de lo mismo: la memoria.

La memoria es donde guardamos los datos, las instrucciones y los resultados.

El mejor depósito dentro de la memoria son las variables, que pueden ser numéricas y alfanuméricas.

Las numéricas pueden ser de dos tipos (ver manual del MSX): ENTERAS que llevan detrás del nombre el signo %, y DECIMALES, que no llevan nada.

DECIMALES: PACO, PEPE, A1, X

ENTERAS: PACO%, PEPE%, A1%, X%

Ejecuta esos programas para comprobar cómo funcionan

12

```
10 REM "AREA DEL CUADRADO"
20 LADO=50
30 AREA=LADO^2
40 PRINT"EL AREA DEL CUADRADO ES=";AR
   EA
50 END
```

13

```
10 REM EJEMPLO DE LOCATE
20 SCREEN 0:COLOR 15,4,4:CLS
30 FOR R=1 TO 40
40 FOR M=1 TO 300:NEXT M
50 LOCATE 16,10:PRINT R
60 NEXT R
70 END
```

14

```
1Ø CLS
2Ø LOCATE 8,9:PRINT "PRUEBA"
3Ø LOCATE 2Ø,5:PRINT "UNA"
4Ø LOCATE 17,13:PRINT "LOCATE"
5Ø LOCATE 12,13:PRINT "CON"
6Ø LOCATE 15,9:PRINT "DE ESCRITURA"
7Ø LOCATE 12,5:PRINT "ESTO ES"
```


15

```
10 REM CRUCES
20 SCREEN 2:COLOR 15,4,4:CLS
30 X=0:X1=255:Y=1:Y1=192:A=1:B=1
40 X=X+A:Y=Y+A:X1=X1-B:Y1=Y1-B
50 IF X>230 THEN 110
60 PSET(X,Y),15
70 PSET(X,Y1),15
80 PSET(X1,Y1),15
90 PSET(X1,Y),15
100 GOTO 40
110 GOTO 110
```

16

```
10 REM CURVAS
20 SCREEN 2:COLOR 1,1,1:CLS
30 FOR J=1 TO 3 STEP .5
40 FOR I=0 TO 510
50 PSET(I/2,(120+80*SIN(I*I/10000))*
(J*J/10)),15
60 NEXT I,J
70 GOTO 70
```

17

```
10 REM PRINCIPIO
20 COLOR15,15,15:SCREEN3
30 X1=4:Y1=4
40 X=X+X1:Y=Y+Y1
50 IF X>255 THEN X1=-4:GOTO40
60 IF X<0 THEN X1=4:GOTO40
70 IF Y>191 THEN Y1=-4:GOTO40
80 IF Y<0 THEN Y1=4:GOTO40
90 C%=1+(14*RND(1))
100 PSET(X,Y),C%:GOTO40
```

Las alfanuméricas llevan siempre, detrás del nombre el signo $ de dólar.

PACO$, PEPE$, CASA$.



No sé si me queda algo más que decirte por hoy. ¡Ah sí! HARDWARE es la parte tangible, lo que se puede tocar: la máquina.

HARDWARE, la cabeza. SOFTWARE: los pensamientos.

Ahí te dejamos algunas prácticas para que te entretengas. Pronto empezaremos con las instrucciones de BASIC, y las prácticas serán más interesantes.

Ejecuta ese programa.

10 PRINT fre (0)

Te da 28803.

Si te fijas, encima del primer OK dice;

28815 BYTES FREE

Siempre que quieras saber los K de memoria RAM de que dispones para hacer tus programas, tecleando PRINT FRE (0) obtendrás la respuesta. En este momento a tu ordenador le quedan libres 28803 BYTES.

```
  28815
- 28803
  -----
     12
```

12 bytes has consumido en ese programilla. Has gastado memoria.

18

```
10 REM REBOTE DE PELOTAS Y CONTROL DE
   CHOQUES
20 ON SPRITE GOSUB 220
30 SCREEN 2:COLOR 15,4,4:CLS
40 LINE(10,10)-(240,180),15,B
50 GOSUB 150
60 A=2:B=2:Y=10:X=10:X1=232:Y1=172
70 X=X+A:X1=X1+-A
80 Y=Y+B:Y1=Y1+-B
90 IF X<12 OR X>232 THEN A=-A
100 IF Y<12 OR Y>172 THEN B=-B
110 PUT SPRITE 0,(X,Y),15,0
120 PUT SPRITE 1,(X1,Y1),11,0
130 SPRITE ON
```

```
140 GOTO 70
150 DATA 24,60,126,126,60,24,0,0
160 RESTORE 150:S$=""
170 FOR K%=1 TO 8
180 READ A:S$=S$+CHR$(A)
190 NEXT K%
200 SPRITE$(0)=S$
210 RETURN
220 FOR R=1 TO 20 STEP 2
230 CIRCLE(X,Y),R,15,,,1.4
240 CIRCLE(X,Y),R,4,,,1.4
250 NEXT R
260 X=12:Y=12:X1=232:Y1=172
270 SPRITE OFF:RETURN
```

Apéndice CSAVE CLOAD

Copia ese programa:

```
 5– REM "EJEM 1"
10– ? "Esto es un ejemplo"
20– ? "de grabación de un"
30– ? "programa en caset"
40– ? "con la instrucción"
50– ? "[ ]C SAVE[ ]"
60– ? END
```

Ejecútalo para ver que funciona.

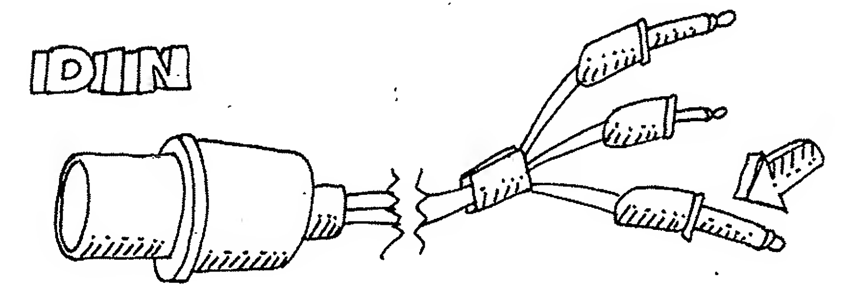
1– Introduce una cinta virgen en tu caset.

2– Ahora pulsa PLAY, y para el caset cuando hayan transcurrido unos 10 segundos.

3– Toma el cable de caset, y de las tres clavijas, introduce en LINE OUT o MIC de tu caset la clavija que no utilizas para hacer el curso. Y la DIN donde ya sabes.

(En caso de que por la pantalla te aparezca algún mensaje de error, acude al manual de tu MSX y busca en el apartado Mensajes de error los siguientes:

DEVICE I/O ERROR
VERIFY ERROR



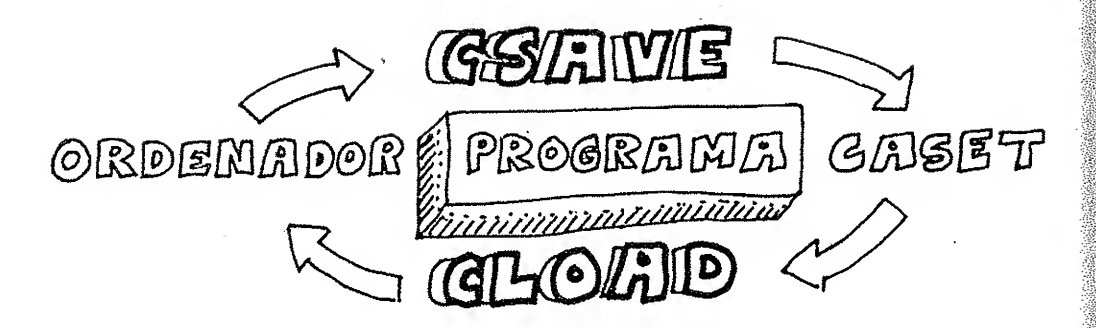
4– Teclea la instrucción CSAVE (que significa "guardar en caset") y luego, entre comillas, escribe el nombre del programa. CSAVE "EJEMPLO".

5– Pulsa RECORD-PLAY en tu caset.

6– Pulsa RETURN

7– Cuando en la pantalla aparezca OK es que el programa ya está grabado en la cinta.

Veamos si es verdad.



1– Extrae la clavija de CSAVE.

2– Introduce la clavija de CLOAD. (La que utilizas para hacer el curso).

3– Rebobina la caset con REWIND.

4– Teclea CLOAD "EJEMPLO". RETURN

5– Pulsa PLAY del caset.

6– En la pantalla aparecerá:

   1– FOUND : EJEMPLO
   2– OK
   ☐

7– Operación concluída.

8– Pulsa F-5

Has grabado en el caset: CSAVE "NOMBRE", y has cargado al ordenador: CLOAD "NOMBRE".

Teclea NEW RETURN.

Copia y ejecuta ese programa.

```
 5–REM "EJEM 2"
10–? "Ejemplo de utilización"
20–? "del comando de búsqueda"
30–? "▭ SKIP ▭"
40–END
```

1– Rebobina la cinta con REWIND.

2– Teclea CLOAD RETURN.

3– Pulsa PLAY del caset.

Como verás, el ordenador te dice que ha cargado EJEM 1, pero no ha cargado EJEM 2.

Cuando se teclea CLOAD sin especificar nombre, el ordenador cargará el primer programa que se encuentre en la cinta: sólo el primero.

1– NEW RETURN

2– Rebobina la cinta con REWIND.

3– Teclea CLOAD RETURN

4– Pulsa el PLAY de caset.

El ordenador encuentra el primer programa, y lo carga.

1– NEW RETURN

2– Rebobina.

3– CLOAD "Ejem 2" RETURN

4– Pulsa PLAY.

Como verás, aparece en la pantalla:

SKIP : EJEM 1

pero el ordenador se lo salta, no lo carga, y sigue buscando "EJEM 2", hasta que lo encuentra.

A partir de hoy, te recomiendo que vayas guardando en un caset todos los programas que te interesen. Recuerda llevar también un cuaderno en el que puedas ir anotando los nombres de los programas, o cualquier otro apunte de interés.

Lección 10

Por fin hemos llegado al BASIC. Hoy vamos a iniciar la parte más práctica del curso. Desde hoy, cada día estudiaremos alguna de las más importantes instrucciones de programación. Hoy vamos a presentarte PRINT y también hablaremos de algo que ya conoces: las variables.

AVANCE REPASO

1. VARIABLE: Es un lugar de la memoria, con nombre, en el que guardamos datos que vamos a utilizar en el programa.

2. VARIABLES: Pueden ser Numéricas o Alfanuméricas.

3. VARIABLES NUMERICAS: Son aquellas en las que guardamos datos numerales para realizar operaciones de cálculo. Si al final les ponemos el signo % (ejemplo PA%) nos redondearán los decimales : sólo nos darán números enteros.

4. VARIABLES NUMERICAS: También llamadas "DE CADENA", su característica más importante es que al final llevan el signo de $ dólar (ej. A$), y los datos tienen que ir siempre, sean números, letras o símbolos, entre comillas. (ej. A$ = "Los 3 mosqueteros").

5. HARDWARE: El ordenador y todos los aparatos que se conectan a él como auxiliares.

6. SOFTWARE: Los programas, la parte no tangible del ordenador.

7. MEMORIA ROM: Una pieza que lleva grabadas una serie de instrucciones fundamentales para el funcionamiento del ordenador, y cuyo contenido nunca se borra.

8. MEMORIA RAM: La pieza en que tú vas a guardar los programas, variables, datos, etc., Se borra con NEW, o cuando se apaga el ordenador. También se llama memoria interna.

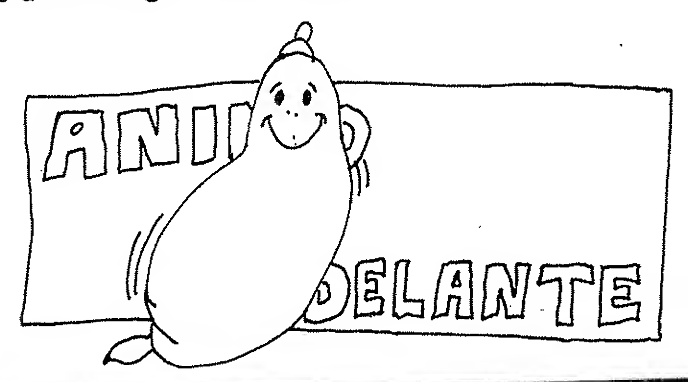
9. MEMORIA EXTERNA: La que está fuera del ordenador. También se llama memoria auxiliar

10. PERIFERICOS: Dispositivos auxiliares del ordenador: joystick, terminal, teclado, impresora, unidad de disco, caset, etc.

11. FRE (Ø): Es una FUNCION. Nos informa de la memoria RAM disponible en ese momento en el ordenador (Formato: PRINT FRE (Ø).

12. FUNCION: Es un tipo de instrucción. Su característica fundamental es que siempre tienen que depender de una instrucción. Es como un adjetivo, que siempre tiene que aparecer asociado a un sustantivo. Una función sólo actúa con la ayuda de una instrucción. (ej. FRE o CHR$ (X) pueden actuar con PRINT).

13. CHR$: Es una función que nos muestra los caracteres del Código ASCII (de Ø a 255).



y ahora presta mucha atención

Ya conoces el camino que recorre un dato desde el teclado hasta que te sale por la pantalla en forma de resultado.

El ordenador funciona con programas.

Hasta ahora has hecho algunas prácticas de BASIC, pero un poco desordenadamente. Ahora vamos a explicarte las instrucciones y funciones más importantes, y su MODO DE ACTUACION.

¿Está listo tu **MSX**? Un ordenador tiene dos modos de actuación.



MODO PROGRAMA

La diferencia está en el número de línea. Modo programa es cuando le introducimos al ordenador un programa, con líneas, cada una de las cuales empieza por un número de línea.

Modo directo es cuando le introducimos instrucciones y datos, pero sin formar programas.

MODO DIRECTO	MODO PROGRAMA
COLOR 3 , 5	10 COLOR 3 , 5 RUN
RETURN	RETURN

La mayor parte de las instrucciones BASIC puede actuar de ambos modos.

Vamos con PRINT. PRINT imprime en la pantalla. PRINT te muestra por la pantalla lo que tú ordenas. Copia y ejecuta ese programa.


```
10  PRINT 1234
15  PRINT
20  PRINT 1 * 2 * 3 * 4
25  PRINT
30  PRINT SORPRESA
35  PRINT
40  PRINT "CORRECTO"
45  PRINT
50  END
```

¿Los has ejecutado?

En ese programa están contenidas las principales actuaciones de PRINT.



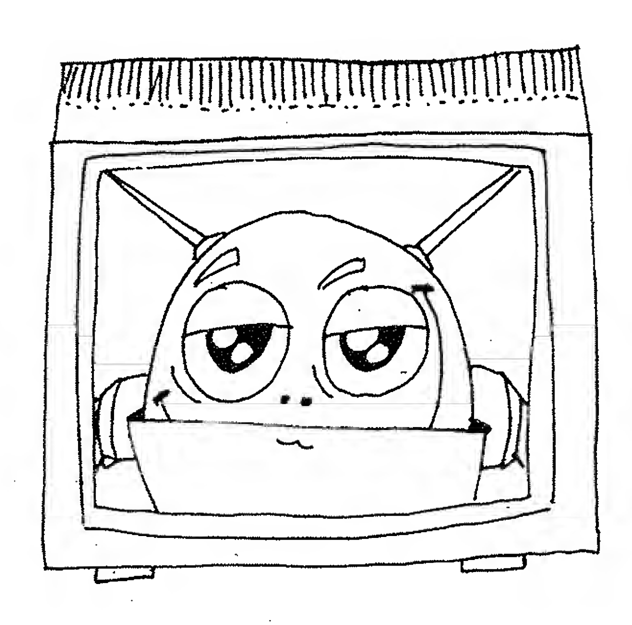
Línea 10: PRINT imprime cualquier constante numeral que no vaya entre comillas.

Línea 20: PRINT muestra el resultado de una operación de cálculo : 24.

Línea 3Ø: PRINT no imprime ninguna constante literal que no tiene que ir entre comillas, por eso imprime Ø.

Línea 4Ø: PRINT imprime, al pie de la letra cualquier constante alfanumérica, que tiene que ir entre comillas, claro.

Líneas 15, 25, 35, 45: PRINT sin ningún mensaje detrás imprime una línea en blanco.



Vamos a ver cómo se porta con las variables.
Copia y ejecuta ese programa.


```
1Ø  A = 1234
2Ø  B = 1 * 2 * 3 * 4
3Ø  C = SORPRESA
4Ø  D$ = "CORRECTO"
5Ø  PRINT A
6Ø  PRINT B
7Ø  PRINT C
8Ø  PRINT D$
9Ø  END
```

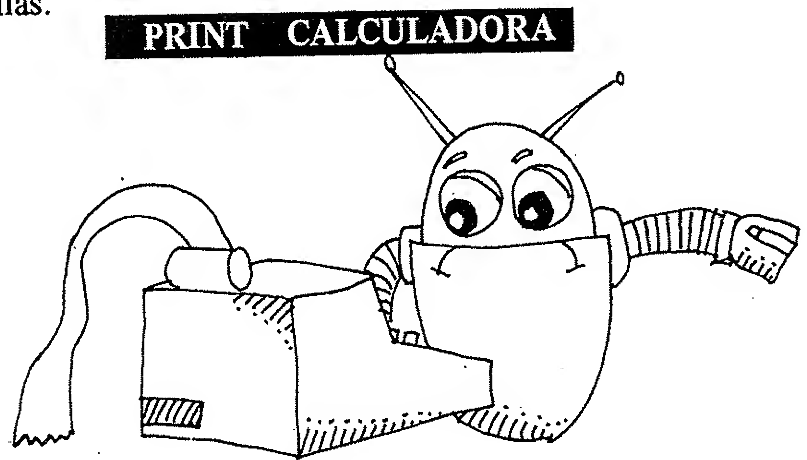
El resultado es el mismo que la otra vez. Lo que tienes que tener siempre presente es que al ordenador hay que darle los datos con claridad, diciéndole si son NUMERICOS (sin comillas) o ALFANUMERICOS (entre comillas).

PRINT imprime lo que le ordenemos.

PRINT IMPRESORA

PRINT copia todo lo que le pongamos entre comillas.

PRINT CALCULADORA



Pero PRINT tiene otra importantísima actuación. El ordenador es, ante todo, una calculadora. Recuerda que el ordenador, en su interior, trabaja con unos y ceros (código binario). El ordenador sabe resolver desde la operación más simple, hasta el cálculo más complejo que tú seas capaz de plantearle.

PRINT sirve para exponer los resultados de todos esos cálculos.

NEW RETURN. SHIFT CLR

```
10 A = 50 : B = 30 : C = 20
20 D = A + B + C
```

Copia y ejecuta ese programa.

¿Por qué no te ha dado el resultado?

Sencillamente porque no se lo has ordenado. En la línea 10 le has ordenado que introduzca o asigne valor a las variables A, B y C.

En la línea 20 le dices que asigne a D el valor de A + B + C. Es decir, le dices que sume A + B + C.

¡Eso es lo que ha hecho el ordenador! ¿Entonces por qué no sale en la pantalla 100 que es el resultado de esa operación? No sale porque no se lo has ordenado.

Copia esta línea.

```
30 PRINT D
```

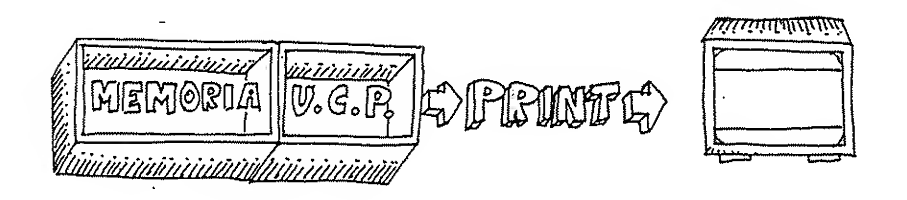

Ejecútalo.

Ahora sí

```
RUN
   100
OK
   □
```

PRINT sirve para mostrar por la pantalla los resultados de las operaciones que hace el ordenador.

PRINT imprime, muestra por la pantalla, los resultados de los procesos que se realizan en la UCP del ordenador.



PRINT permite que se vean los resultados, por ello decimos que PRINT es la calculadora de tu MSX.

Pero ¿quién hace los cálculos? Los cálculos los hacen los OPERADORES ARITMETICOS:

1º	∧	Potenciación
2º	*	Multiplicación
	/	División
3º	+	Suma
	–	Resta

Tengo que decirte que el ordenador tiene un orden establecido para todo. Del mismo modo que para hacer un programa empieza por la línea de número más bajo y acaba por la de número más alto, para hacer las operaciones, tiene un orden:

1º Hace las potenciaciones.
5 ∧ 2 = 25

2º Hace las multiplicaciones
5 * 2 = 25
y divisiones, indistintamente
6 / 3 = 2

3º Hace las sumas
3 + 9 = 12
y restas
15 – 10 = 5

Ejecuta y observa este ejemplo.

```
PRINT 2 + 3 * 5
```


¿Cuál crees que será el resultado de esa operación?

1ª 3 * 5 = 15
2ª 15 + 2 = 17

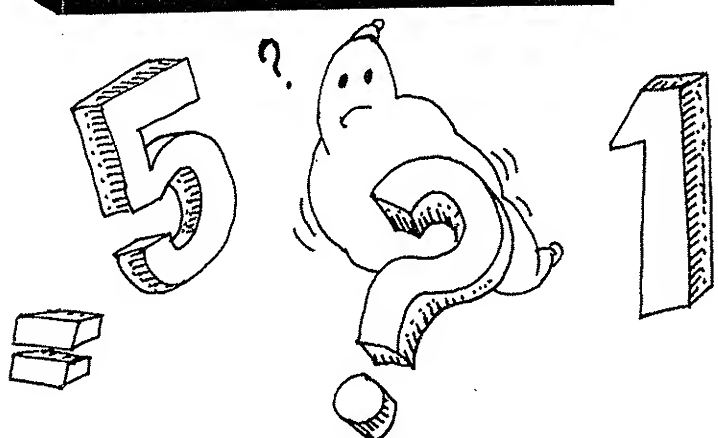
Vamos a hacer un ejercicio. Calcula el resultado de esas operaciones y luego comprueba con el ordenador en modo directo si has acertado.

```
PRINT 2 + 8 * 5 – 3 = ____________
PRINT 8 * 3 / 6 – 6 + 2 = ____________
PRINT 10 * 2 ∧ 2 / 4 = ____________
PRINT –5 + 10 ∧ 3 * 8 / 5 ____________
```

Esto no es difícil aunque si requiere abundantes prácticas por tu parte.

Copia y ejecuta ese programa.

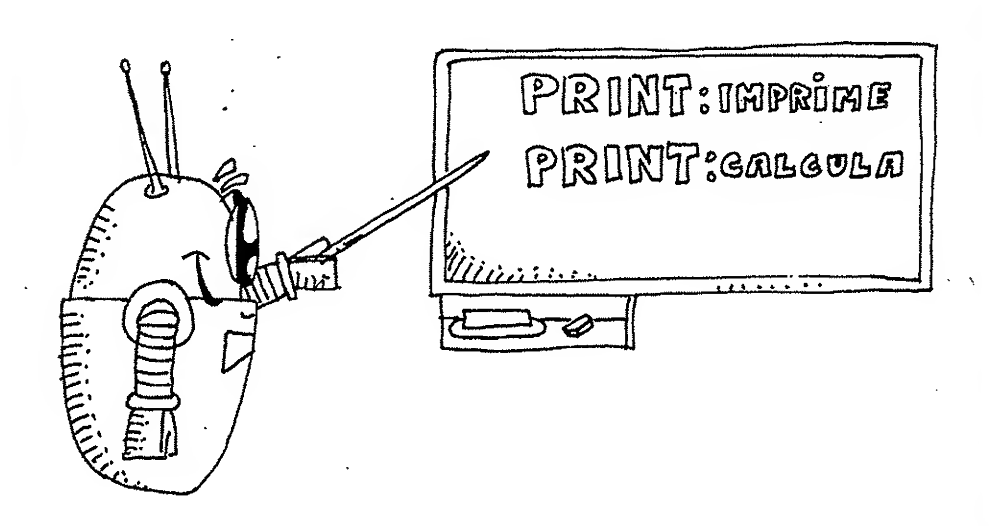
```
10 A = 3 : B = 8 : C = 4
20 D = 9 : E = 7 : F = 5
30 R1 = D E – A C F 2
40 R2 : A – B C 3 D – E F
50 ? R1
60 ? R2
```



Otra cosa que debes conocer sobre PRINT CALCULADORA, son los PARENTESIS. Los paréntesis en matemáticas tienen un orden para ejecutarse. Los paréntesis pueden alterar ese orden. De hecho lo alteran. **Cuando un ordenador encuentra una operación, lo primero que hace es resolver lo que va entre paréntesis, empezando de izquierda a derecha; luego sigue con el orden que ya conoces.**

PRINT (5 + 3) * 8 ∧ 4 / 9 * (3 – 13) * 8 =
1º 4º 3º 5º 6º 2º 7º

1º Paréntesis
2º Potenciación
3º Multiplicación y división (empezando por la izquierda).
4º Suma y Resta (empezando por la izquierda).



Todavía no hemos terminado con PRINT. PRINT imprime : PRINT muestra resultados.

```
10 PRINT "CAMI" ;
20 PRINT "SETA"
```

Ejecútalo

```
10 PRINT "CAMI", "SETA"
```

Ejecútalo

LA COMA SEPARA
EL PUNTO Y COMA ENGANCHA



RAIZ CUADRADA

Pero todavía queda más. ¿Cómo se hace una raíz cuadrada? Hay dos modos:

PRINT 625 ∧ 0.5 = 25

Hemos elevado 625 a 0.5. (en BASIC, para poner decimales utilizamos el punto, no la coma).

Otro modo es con una función : SQR (N).

PRINT SQR (625)

PRACTICAS TAREAS PROGRAMAS

El resto de la lección vas a dedicarla a ejecutar programas, y a confeccionar otros por tu cuenta.

19

```
10 REM "AREA DEL CIRCULO"
20 PI=3.141592A
30 RADIO=18
40 AREA=PI*RADIO^2
50 PRINT"EL AREA DEL CIRCULO ES ";ARE
A
60 END
```

20

```
10 REM "REGLA DE TRES"
20 A=5
30 B=100
40 C=50
50 R=B*C/A
60 PRINT A;" ES A ";B;" COMO ";C;" ES
   A ";R
70 END
```

21

```
<<<<<<PROGRAMAS LECCION 12>>>

10 REM << AREA DEL TRIANGULO>>>
20 REM B=BASE H=ALTURA A=AREA
30 B=20
40 H=30
50 A=B*H/2
60 PRINT "EL AREA ES = ";A
70 END
```


22

```
INSTRUCCION "INT"

MODIFICA LA LINEA 50 DEL PROGRAMA
ANTERIOR ASI:

50 PRINT X;",";INT(X)
```

23

```
INSTRUCCION "SQR"

10 CLS

20 INPUT "DE QUE NUMERO QUIERES LA RA
IZ CUADRADA";A

30 PRINT "LA RAIZ CUADRADA DE";A;" ES
 IGUAL A";SQR(A)
```

24

```
INSTRUCCION "ASC"

10 CLS

20 INPUT "DE QUE QUE CARACTER QUIERES
 EL VALOR ASCII";A$

30 PRINT "SU VALOR ASCII ES";ASC(A$)
```

25

```
10 REM TUNEL
20 SCREEN 2:COLOR 15,4,4:CLS
30 A$="RDLU"
40 FOR R=1 TO 3
50 FOR S=4 TO 250 STEP 4
60 X=X+1:Y=Y+1
70 DRAW"C15S"+STR$(S)+"BM"+STR$(X)+",
   "+STR$(Y)+A$
80 NEXT S
90 NEXT R
100 GOTO 100
```

26

```
10 REM EJEMPLO DE DRAW
20 SCREEN 2:COLOR 15,4,4:CLS
30 DRAW"BM40,100E20F10NL20F10BR5U20F2
0U20BR5R20L10D20BR20R20U20L20D20BR25U
20F20U20BR5R6L3D20L3R6BR5U20R20D20L20
BR20D4L137H3"
40 GOTO 40
```


Lección 12

¿Qué te ha parecido PRINT? Es una de las instrucciones más importantes del BASIC, y tienes que manejarla con destreza. ¡Y cuidado con los SYNTAX ERROR: Solo haciendo prácticas aprenderás a programar!



Hoy vamos a hablar de una instrucción muy relacionada con las variables (como todo el BASIC) : INPUT.

Pero antes. . .

AVANCE REPASO

1– MODOS DE FUNCIONAMIENTO. El ordenador puede funcionar en modo directo o en modo programa. En MODO DIRECTO las instrucciones se ejecutan con **RETURN**. En MODO PROGRAMA las instrucciones forman parte de líneas numeradas, y hay que ejecutarlas con RUN **RETURN**, o **F5**.

2– FUNCION. Es una instrucción que actúa dependiendo de otra principal.

3– SQR (N). Es la función que sirve para hacer raíces cuadradas. Va asociada a PRINT. (La raíz cuadrada también se puede obtener elevando el número en cuestión a 0.5).

4– PRINT. Sirve para imprimir cadenas de caracteres, y para mostrar el resultado de las operaciones de cálculo.

5– PUNTO y COMA. Une cadenas de caracteres.

6– COMA. Separa cadenas de caracteres.

7– DOS PUNTOS. Permite introducir más de una instrucción en una misma línea.

ej. 1Ø LET A = 2Ø : LET B = 3Ø: LET A = 5Ø

8– REM. Es una instrucción que sirve para poner textos, aclaraciones, nombres, en los programas. Los mensajes de REM no aparecerán en la ejecución del programa. Solo aparecerán cuando se recupera la memoria, el listado del programa, con la instrucción LIST. (REM puede ponerse también con el signo ´).

9– OPERACIONES DE CALCULO. Se realizan con los operadores aritméticos:

^	*	/	+	–

10– ORDEN DE LOS OPERADORES. 1º) Lo que vaya entre paréntesis. 2º) Potenciación. 3º) Multiplicación y división. 4º) Suma y resta.

11– ? . Es equivalente a PRINT.

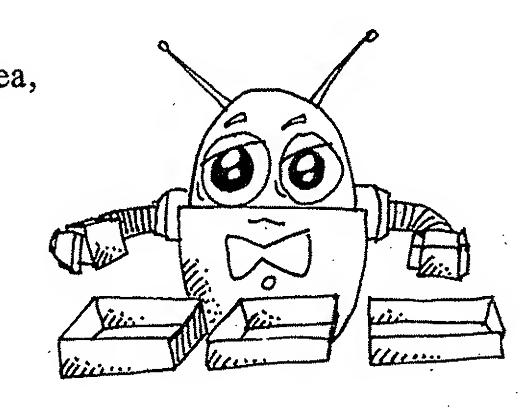
ASIGNACION

Se acabó el repaso. Vamos con el tema de hoy, que es la asignación. **La asignación es la operación mediante la cual damos valor a una variable.**

ej. A = 5, B$ = "HOLA", C$ = "1985"

Ponemos el nombre de la variable, y al lado el signo de igual y el valor que asignamos a la variable.

Y tú dirás,
si en cada línea
va un número de línea,
una instrucción,
y algún dato,
¿cuál es
la instrucción
que sirve para
hacer la
asignación?



1Ø A = 2Ø

¿Está mal esa línea? No. Lo que ocurre es que la instrucción de la asignación se usa tanto, es tan necesaria,

que el BASIC ha previsto que, si no se quiere, no se escriba: se sobreentiende. La instrucción de la asignación es LET. La forma correcta de asignar valor a las variables es:

```
1Ø LET A = 2Ø
```

Pero LET puede no ponerse. Con LET asignamos valor a las variables.

Con INPUT asignamos también valor a las variables, pero desde fuera del programa.

```
1Ø INPUT N
```

Como ves, N no tiene valor asignado. Cuando el ordenador encuentra INPUT espera que alguien se acerque al teclado y asigne valor a N a través del teclado.

LET e INPUT son las instrucciones que sirven para asignar valor a las variables.

Copia esa línea.

```
1Ø INPUT N
```

La variable de INPUT es numérica, por tanto habrá que asignarle valor numérico. Pulsar **F5** Te aparecerá en la pantalla el signo ? . Ahora teclea HOLA y pulsa **RETURN**

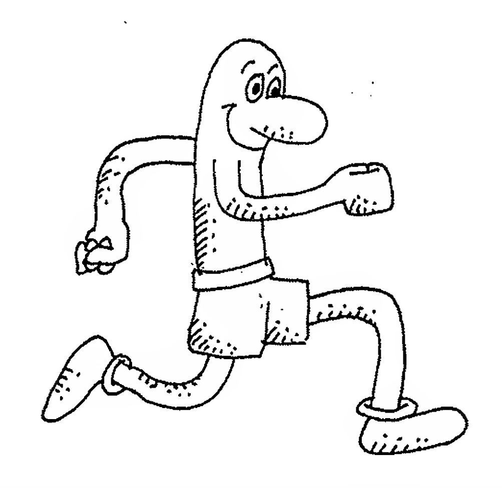
En la pantalla te aparece

REDO FROM START

Este mensaje de error quiere decir que has asignado un dato alfanumérico a una variable numérica.

Bien. No lo olvides.

Ahora vamos a la práctica.



PRACTICAS PROGRAMAS TAREAS

Si quieres ser un programador, tienes que programar. Haz, al menos, cinco programas para calcular superficies, áreas, altura, volumen de figuras geométricas, o para aplicar, cualquier tipo de fórmula.

27

```
5 SCREEN 1
10 REM <<< CAMBIO DE COLOR >>>
20 INPUT"COLOR PAPEL";CP%
30 INPUT"COLOR TINTA";CT%
40 INPUT"COLOR BORDE";CB%
50 IF CP%>15 OR CP%<1 OR CT%>15 OR CT
%<1 OR CB%>15 OR CB%<1 THEN PRINT "DA
TOS INCORRECTOS, REPITE LOS DATOS":GO
TO 20
60 COLOR CT%,CP%,CB%:CLS
```

28

```
10 REM PREGUNTAS
20 INPUT "Que edad tienes";A
30 INPUT "Como te llamas";B$
40 CLS
50 PRINT "Tu nombre es ";B$
60 PRINT "y tienes";A;"años"
70 END
```

```
5 SCREEN 1
10 REM <<< AMPLIACION A CAMBIO DE COL
OR >>>
20 INPUT"COLOR PAPEL";CP%
30 INPUT"COLOR TINTA";CT%
40 INPUT"COLOR BORDE";CB%
50 IF CP%>15 OR CP%<1 OR CT%>15 OR CT
%<1 OR CB%>15 OR CB%<1 THEN PRINT "DA
TOS INCORRECTOS, REPITE LOS DATOS":GO
TO 20
60 COLOR CT%,CP%,CB%:CLS
70 PRINT"COLOR PAPEL ";CP%
80 PRINT"COLOR TINTA ";CT%
90 PRINT"COLOR BORDE ";CB%
100 GOTO 20
```

29

30

```
10 REM EJEMPLO DE LOCATE
15 SCREEN0:COLOR 15,4,4:CLS
20 LOCATE 0,1:PRINT "LOCATE"
30 LOCATE 1,2:PRINT "LOCATE"
40 LOCATE 2,3:PRINT "LOCATE"
50 LOCATE 3,4:PRINT "LOCATE"
60 LOCATE 4,5:PRINT "LOCATE"
70 LOCATE 5,6:PRINT "LOCATE"
80 LOCATE 6,7:PRINT "LOCATE"
90 LOCATE 7,8:PRINT "LOCATE"
100 LOCATE 8,9:PRINT "LOCATE"
110 LOCATE 9,10:PRINT "LOCATE"
120 LOCATE 10,11:PRINT "LOCATE"
130 LOCATE 11,12:PRINT "LOCATE"
140 LOCATE 12,13:PRINT "LOCATE"
150 LOCATE 13,14:PRINT "LOCATE"
160 LOCATE 14,15:PRINT "LOCATE"
170 LOCATE 15,16:PRINT "LOCATE"
180 LOCATE 16,17:PRINT "LOCATE"
190 LOCATE 17,18:PRINT "LOCATE"
200 LOCATE 18,19:PRINT "LOCATE"
210 LOCATE 19,20:PRINT "LOCATE"
220 LOCATE 20,21:PRINT "LOCATE"
230 LOCATE 21,22:PRINT "LOCATE"
240 LOCATE 22,23:PRINT "LOCATE"
250 LOCATE 23,24:PRINT "LOCATE"
260 END
```

31

```
10 REM EJEMPLO DE DRAW Y ESCALA
20 SCREEN 2:COLOR 15,4,4:CLS
30 FOR S=1 TO 60
40 DRAW"C15S"+STR$(S)+"BM100,90U5R5D5
L5"
50 NEXT S
60 FOR S=60 TO 1 STEP -1
70 DRAW"C4S"+STR$(S)+"BM100,90U5R5D5L
5"
80 NEXT S
90 GOTO 30
```

```
10 REM DESCONTROL DE TECLADO
20 SCREEN 0:COLOR 15,4,4:CLS
30 H=RND(-TIME)
40 A$=INKEY$:IF A$="" THEN 40
50 C=RND(1)*5+1
60 IF ASC(A$)+C>255 THEN 50
70 B=ASC(A$)+C
80 PRINT CHR$(B)
90 GOTO 40
```

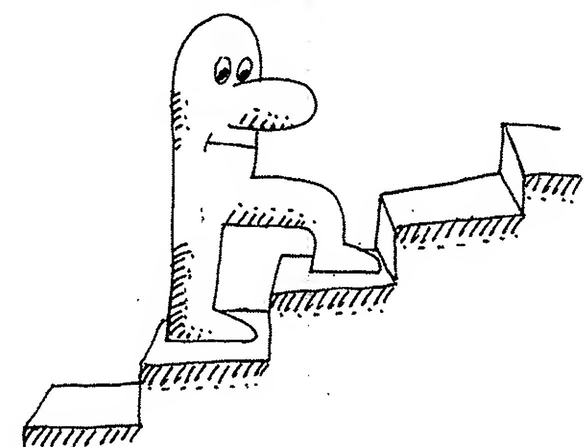
32

```
10 REM <<PANTALLA CIRCULOS >>
20 SCREEN 2:COLOR 15,4,4:CLS
30 FOR X=6 TO 250 STEP 20
40 FOR Y=6 TO 190 STEP 20
50 CIRCLE(X,Y),10
60 NEXT Y,X
70 GOTO 70
```

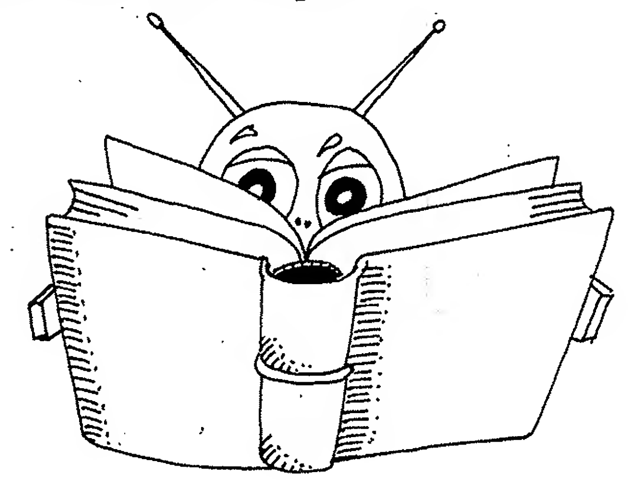
33

Lección 14

Cada vez se enriquece más tu vocabulario informático. Cada vez conoces más palabras del idioma BASIC, de su sintaxis. No obstante te queda mucho por aprender. Hoy daremos otro paso en nuestro camino hacia la comprensión de la programación.



Pero antes vamos a repasar.



AVANCE REPASO

1– INPUT. Es la instrucción que sirve para asignar valor a las variables desde fuera del programa. Su formato:

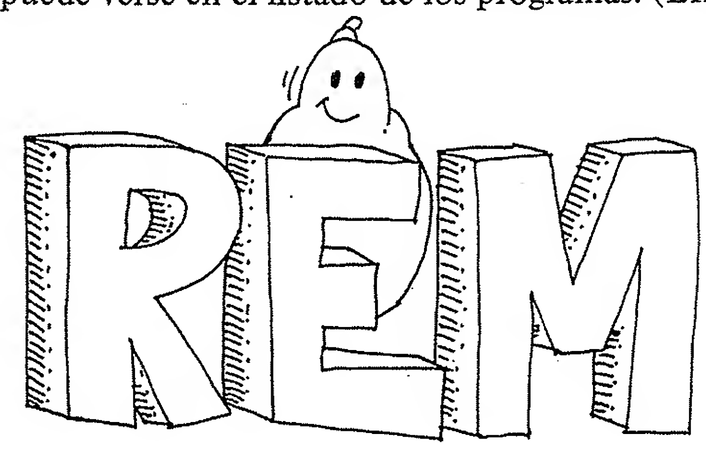
INPUT "MENSAJE" ; VARIABLE (o VARIABLES)

2– REDO FROM START. Es el error que se produce cuando se asigna, con INPUT, valor alfanumérico, a una variable numérica.

3– LET. Sirve para asignar valor a las variables. Su uso es opcional, porque se puede asignar valor simplemente poniendo el nombre de la variable y el signo de igual.

10 A$ = "HOLA"

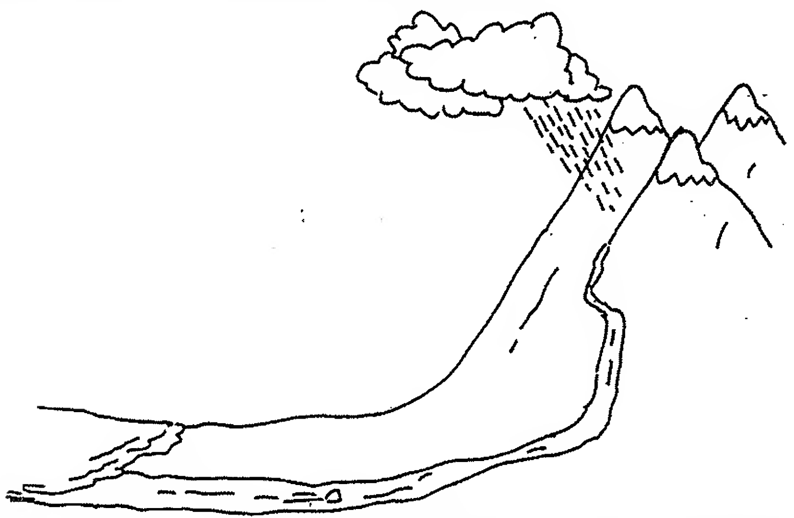
4– REM. Sirve para poner título a los programas, o hacer comentarios en ellos. No sale en pantalla. Solo puede verse en el listado de los programas. (LIST).



FLUJOS

Sin más preámbulos, vamos a poner orden en los programas.

Las nubes llevan agua; llueve; cae agua en las montañas; se forman ríos; los ríos recorren los valles; atraviesan las llanuras, y desembocan en el mar.



Un programa también. Un programa tiene un principio, como los ríos. Con LET e INPUT recibe datos (lluvia), recorre líneas (valles), y finalmente, llega el END, fin (mar).

Un programa tiene principio, desarrollo y fin. Un río fluye, y tiene un caudal, un flujo de agua. Un programa tiene un caudal de instrucciones y datos, que fluyen a lo largo de las líneas, hasta llegar al fin. Un programa empieza en RUN y acaba en END. Y va ordenadamente.

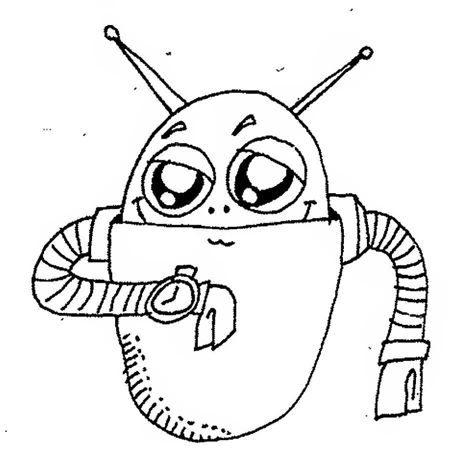
Efectivamente. Si a FOR . . . NEXT no se le pone tarea, hace como GOTO cuando se dirigía sobre su propia línea : bloquea el programa, pero lo bloquea en tanto en cuanto termina de ejecutarse; una vez que ha terminado de ejecutarse, permite que el programa siga su flujo normal.

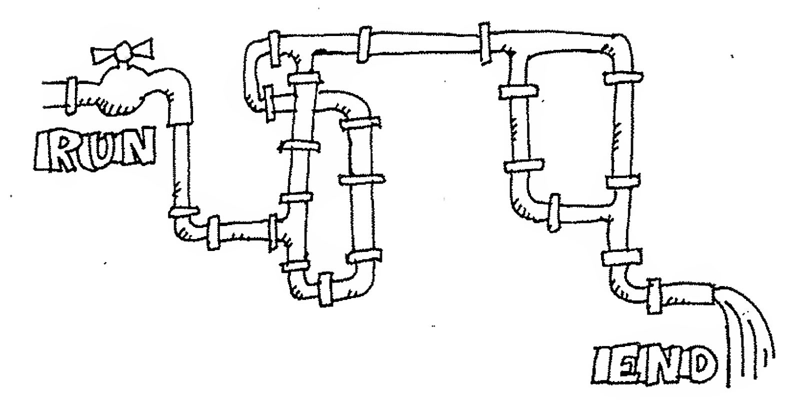
```
10 PRINT "TEMPORIZADOR"
20 FORT = 1 TO 5000: NEXT T
30 PRINT "Se acabó el temporizador"
40 END
```

Ejecuta ese programa.

¿Has visto? EL TEMPORIZADOR te será muy útil en múltiples tareas. Ya lo comprobarás.

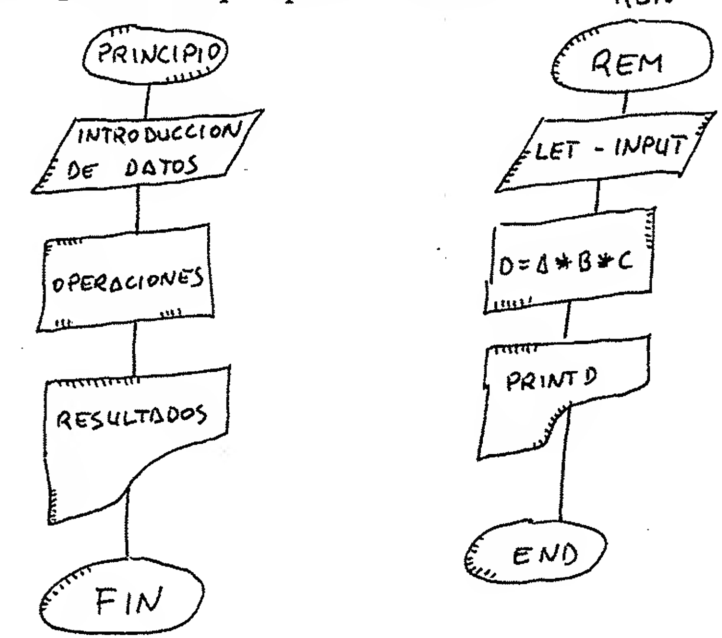
FOR . . . NEXT es un contador controlado, un bucle controlado, un repetidor de tareas, un controlador, un temporizador.





Antes de hacer un programa hemos de tener claro QUE ES LO QUE QUEREMOS HACER. Hemos de planificar, hay que tener claras las ideas, hay que poner orden. . . y para ello vamos a utilizar un ORDINOGRAMA, también llamado ORGANIGRAMA.

Es sencillamente un dibujo para representar de un modo gráfico lo que queremos hacer.



Mira el ejemplo.

```
10 REM programa de ejemplo.
20 INPUT "Dime 3 números" ; N1, N2, N3.
30 B = N1 * N2 * N3
40 PRINT "Resultado = " ; B
50 END
```

Hemos seguido fielmente el orden del Ordinograma. Puede que te parezca una labor innecesaria, pero te aseguro que cuando tengas que enfrentarte a un programa de envergadura no te estorbará.

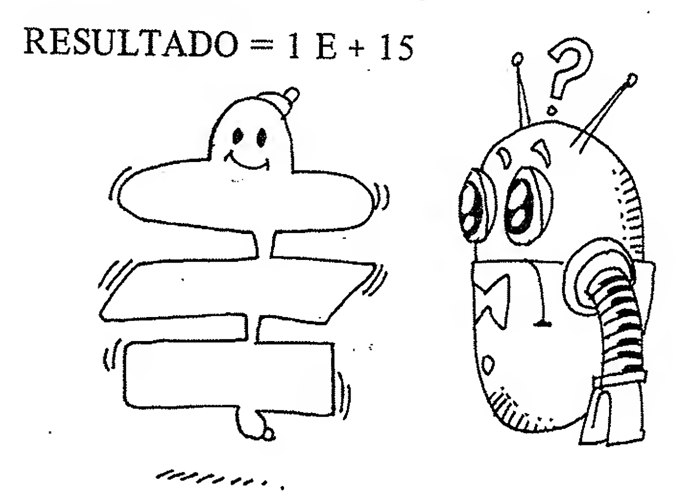
Copia y ejecuta el programa anterior.

Ahora, vuelve a ejecutarlo, y cuando te pida los números con INPUT, introdúcele los siguientes:

100.000, 100.000 y 100.000

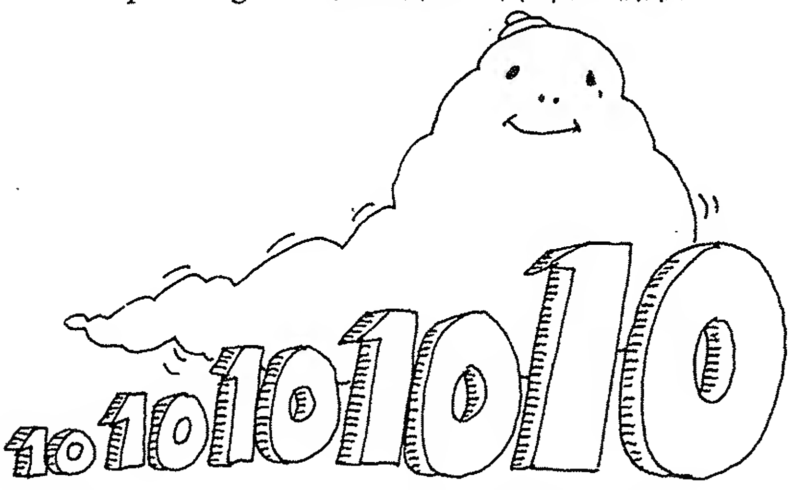
En la pantalla te aparece :

RESULTADO = 1 E + 15

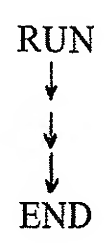


Eso quiere decir que el número resultante es tan grande que el ordenador prefiere escribirlo con NOTACION CIENTIFICA, que es como una abreviatura que se emplea para escribir números muy grandes 1E +15 quiere decir que el número resultante es 10 elevado a15, es decir: 10 * 10 * 10 * 10 * 10 * 10 * 10 * 10 * 10 * 10 * 10 * 10 * 10 * 10 * 10

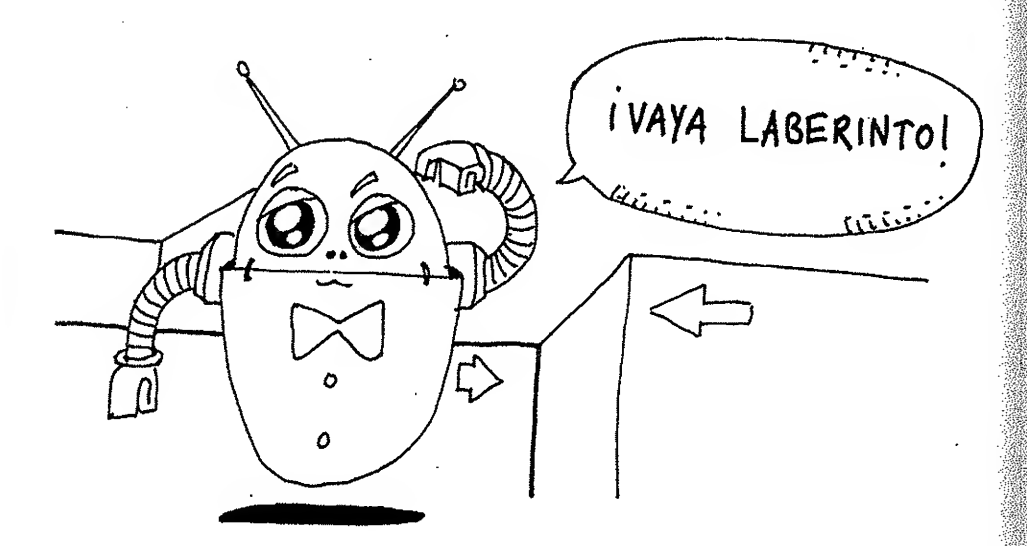
O lo que es igual 1.000.000. 000.000.000.



Bueno, sigamos con el ORDINOGRAMA. Un programa empieza por el principio y se acaba por el final. El controlador, el control, que está en la UCP tiene mucho cuidado de que todo se haga en orden, siguiendo el FLUJO DE CONTROL. El flujo de control normal en un programa es:

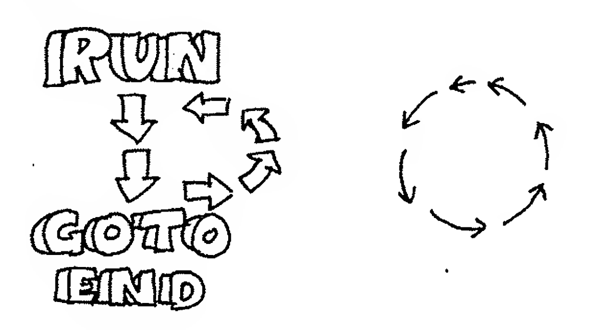


Y esa figura que representa el flujo de control es un DIAGRAMA DE FLUJO, que también te será de gran utilidad a la hora de resolver programas complejos.



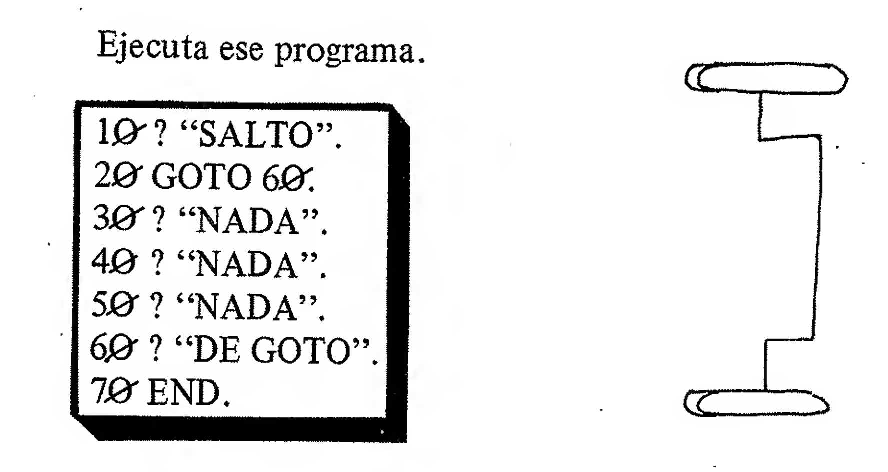
Vamos ahora a abordar el estudio de una instrucción que, precisamente, tiene influencia sobre el flujo de control normal de un programa. Se trata de GOTO.

GOTO, como ya sabes, significa "VE HACIA". Su flujo es circular.



Aunque GOTO no siempre funciona hacia arriba, también puede dirigir el flujo de control hacia líneas de numeración más alta.

Ejecuta ese programa.

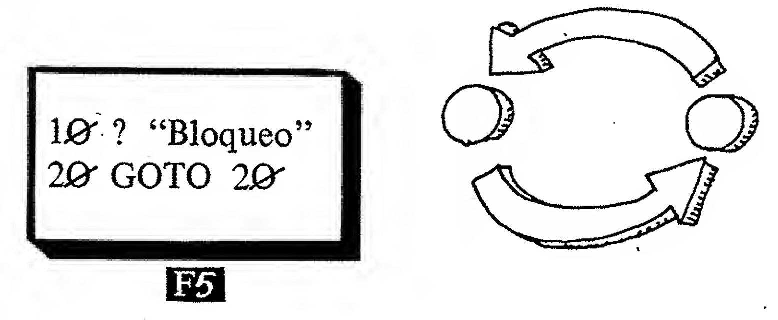


```
10 ? "SALTO".
20 GOTO 60.
30 ? "NADA".
40 ? "NADA".
50 ? "NADA".
60 ? "DE GOTO".
70 END.
```

En este caso GOTO ha funcionado como un salto. Se ha saltado las líneas 30, 40, 50.

GOTO también se puede dirigir sobre su propia línea, y entonces ocurrirá algo lógico.

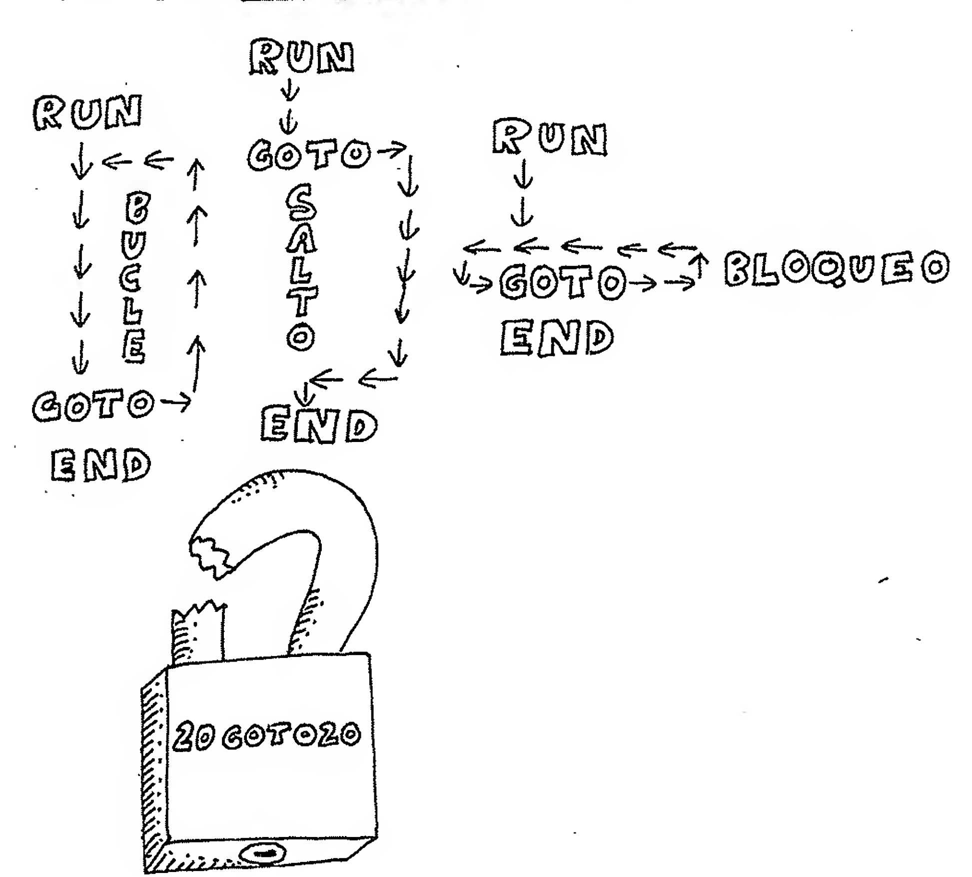
Ejecuta ese programa. (Antes de ejecutar cada programa, recuerda siempre limpiar la memoria del ordenador).



```
10 ? "Bloqueo"
20 GOTO 20
```

F5

Ahora pulsa las teclas que quieras, todas, de tu ordenador: ha quedado bloqueado. Para desbloquear el programa pulsa CTRL STOP. Con ello has roto el bloqueo.



Ahora ejecuta ese programa

```
10 A = 0.
20 A = A + 1.
30 PRINT A.
40 GOTO 20.
```

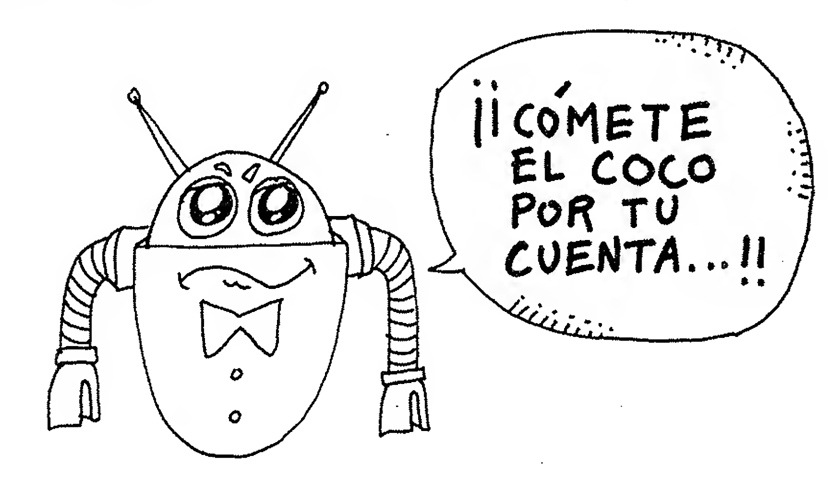

Ya lo conoces, ¿ves cómo funciona? Te aconsejo que te lo aprendas de memoria (sí de memoria).

Ahora vamos a hacer una pequeña disgresión.

Ejecuta ese programa.

```
10 A = 0.
20 A = A + 1.
30 PRINT A.
40 RUN 20.
```

RUN puede dirigir también el flujo de control en las mismas direcciones que GOTO. Ejecuta esos dos programillas y luego. . .



```
10 PRINT "Saltito"
20 RUN 40
30 PRINT "NADA"
40 PRINT "de RUN"
50 END
```

```
10 ? "Bloqueíllo"
20 RUN 20
```

Por hoy está bien. Ahí te dejamos con una tanda de ejercicios.

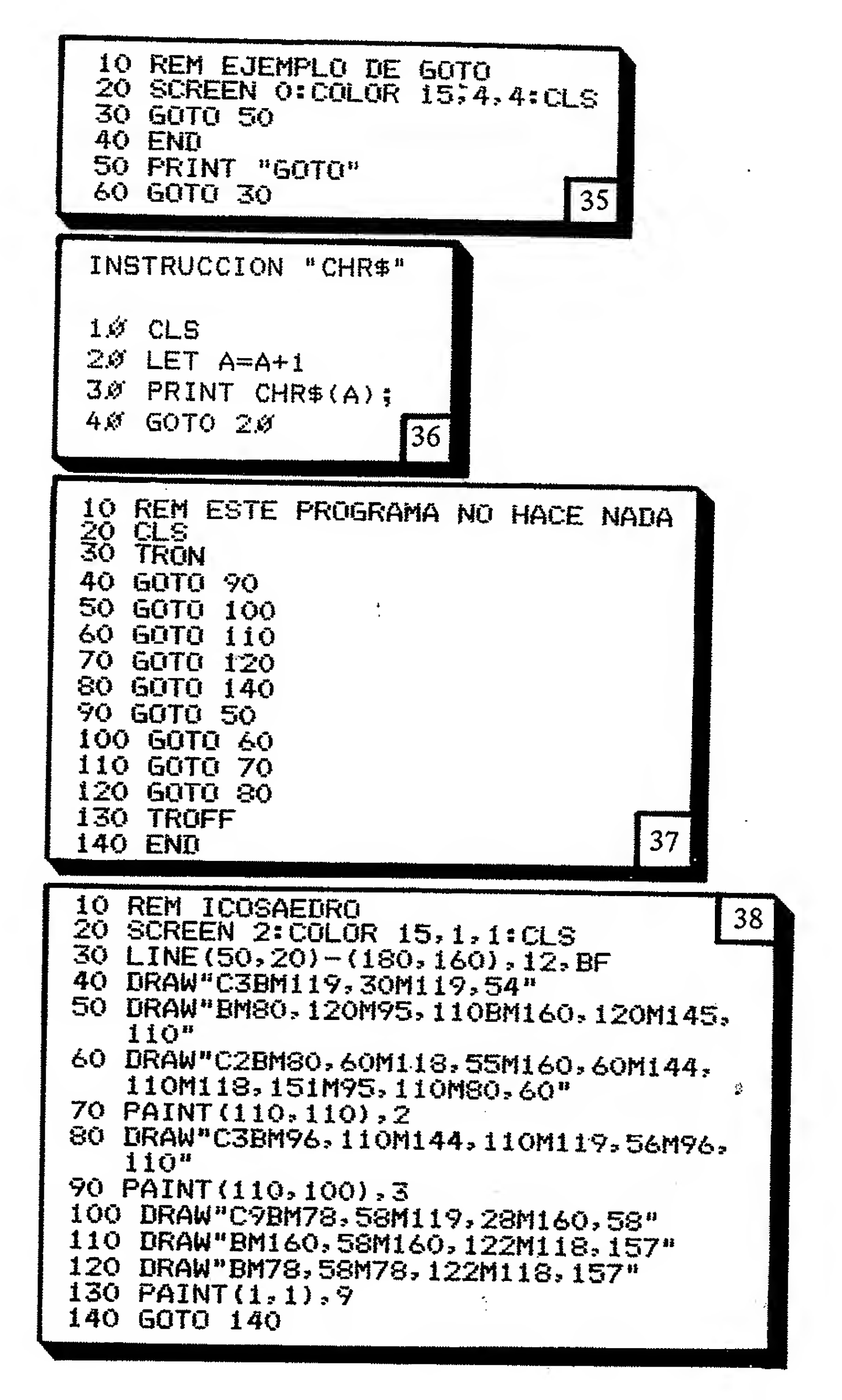
PRACTICAS PROGRAMAS TAREAS

Con PRINT, INPUT, y los operadores aritméticos, trata de resolver los problemas más corrientes de cálculo, referentes a la materia que estés estudiando, o que mejor conozcas.

Tienes que hacer, al menos, 5 programas.

```
10 REM EJEMPLO DE GOTO
20 A=A+1:IF A>3 THEN GOTO 40
30 ON A GOTO 50,60,70
40 END
50 PRINT "GOTO"
60 PRINT "GOTO"
70 PRINT "GOTO"
80 GOTO 20
90 GOTO 90
```

34



```
10 REM EJEMPLO DE GOTO
20 SCREEN 0:COLOR 15;4,4:CLS
30 GOTO 50
40 END
50 PRINT "GOTO"
60 GOTO 30
```

35

```
INSTRUCCION "CHR$"

1Ø CLS
2Ø LET A=A+1
3Ø PRINT CHR$(A);
4Ø GOTO 2Ø
```

36

```
10 REM ESTE PROGRAMA NO HACE NADA
20 CLS
30 TRON
40 GOTO 90
50 GOTO 100
60 GOTO 110
70 GOTO 120
80 GOTO 140
90 GOTO 50
100 GOTO 60
110 GOTO 70
120 GOTO 80
130 TROFF
140 END
```

37

38

```
10 REM ICOSAEDRO
20 SCREEN 2:COLOR 15,1,1:CLS
30 LINE(50,20)-(180,160),12,BF
40 DRAW"C3BM119,30M119,54"
50 DRAW"BM80,120M95,110BM160,120M145,
   110"
60 DRAW"C2BM80,60M118,55M160,60M144,
   110M118,151M95,110M80,60"
70 PAINT(110,110),2
80 DRAW"C3BM96,110M144,110M119,56M96,
   110"
90 PAINT(110,100),3
100 DRAW"C9BM78,58M119,28M160,58"
110 DRAW"BM160,58M160,122M118,157"
120 DRAW"BM78,58M78,122M118,157"
130 PAINT(1,1),9
140 GOTO 140
```

39

```
10 REM INTENTO DE SOMBRAS
20 SCREEN 2:COLOR 15,4,4:CLS
30 DRAW"BM112,40C7NG200R30F200"
35 PAINT(120,100),7
40 DRAW"BM120,60C15BR5U20L13C14NE5U5E
10R10F10D5H5NL20F5C15NL20L13D20R6D6L1
6U6R6"
45 PAINT(120,30),14
50 DRAW"C5BM24,180M84,100M110,120M66,
180M24,180"
55 PAINT(50,160),5
60 CIRCLE(96,110),20,9,,,1.4
65 PAINT(96,110),9
70 CIRCLE(40,100),20,6,,,1.4
75 PAINT(40,100),6
80 CIRCLE(46,180),20,5,,,.2
85 PAINT(46,183),5
90 GOTO 90
```

40

```
10 REM CUADRO
20 SCREEN3
30 LINE(63,85)-(170,128),1,B
40 REM MOVIMIENTO
50 FORX%=63TO170
60 PSET(X%,85),8
70 PSET(X%,85),1
80 NEXTX%
90 FORY%=85TO 128
100 PSET(170,Y%),8
110 PSET(170,Y%),1
120 NEXTY%
130 FORX%=170TO63STEP-1
140 PSET(X%,128),8
150 PSET(X%,128),1
160 NEXTX%
170 FORY%=128TO85STEP-1
180 PSET(63,Y%),8
190 PSET(63,Y%),1
200 NEXTY%
210 GOTO 50
```

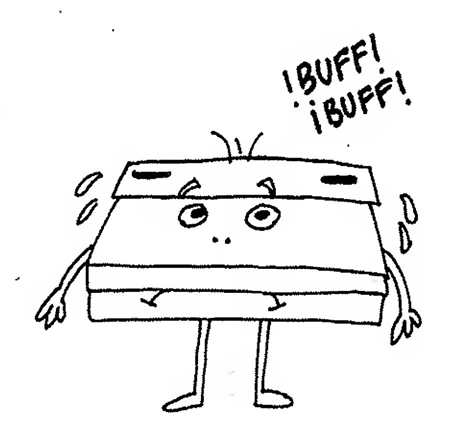


```
10 REM RECUADRAR
20 SCREEN 2:COLOR 15,4,4:CLS
30 A=0:X=0:Y=0:X1=240:Y1=190:X2=0:
   Y2=0
40 X=X+1:PSET(X,Y),15
50 IF X=142 AND Y=94 THEN GOTO 170
60 IF X<X1-A THEN 40
70 A=A+2
80 Y=Y+1:PSET(X,Y),15
90 IF Y<Y1-A THEN 80
100 A=A+2
110 X=X-1:PSET(X,Y),15
120 IF X>X2+A THEN 110
130 A=A+2
140 Y=Y-1:PSET(X,Y),15
150 IF Y>Y2+A THEN 140
160 A=A+2:GOTO 40
170 GOTO 170
```

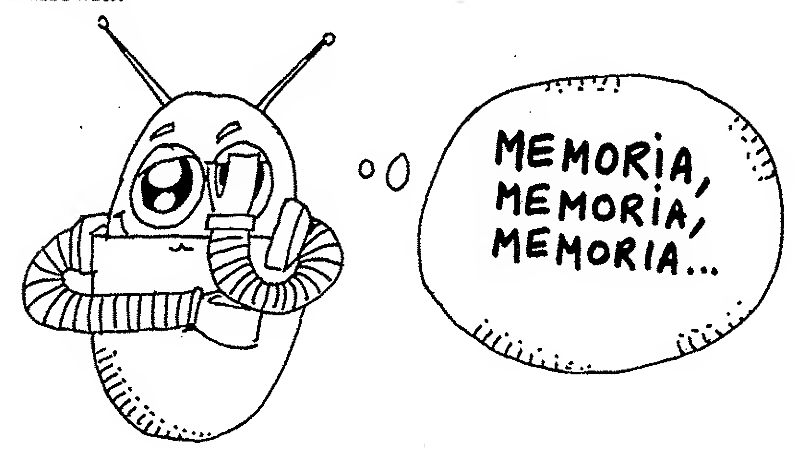
41

Lección 16

Naturalmente.
Ya has ejecutado más de cien programas, o al menos eso espero.
No creas en ningún momento que el BASIC se puede aprender de "oído".
El Basic sólo se aprende trabajando con el ordenador: trabajando. Hay que tener las ideas claras.



Antes de empezar con la lección de hoy hagamos memoria.



AVANCE REPASO

1. END. Es el punto y final de los programas.

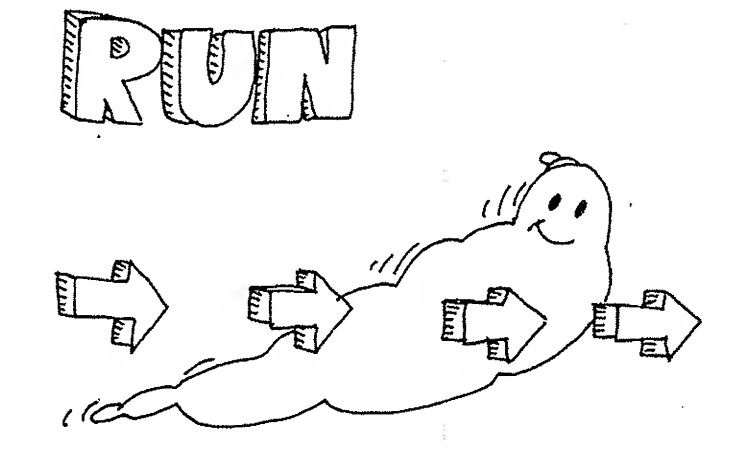
2. ORDINOGRAMA. También llamado Organigrama, es la representación gráfica de una tarea.

3. ALGORITMO. Es el conjunto de instrucciones necesarias para realizar en orden una tarea concreta.

4. DIAGRAMA DE FLUJO. Es la representación gráfica del flujo de un programa.

5. GOTO. Es una instrucción que interrumpe el flujo lineal del programa, remitiéndolo a una línea determinada.

6. RUN. Inicio de flujo de cualquier programa. También puede dirigir el flujo en otras direcciones.



7. SISTEMA BINARIO: Es un sistema de numeración, en el que sólo hay unos y ceros (en el sistema decimal hay del Ø al 9). Como recordarás es el sistema que utiliza el ordenador para desarrollar su lenguaje máquina. Ahí tienes algunos ejemplos de números en sistema binario:

DECIMAL	BINARIO
1	ØØØØØØØ1
2	ØØØØØØ1Ø
1Ø	ØØØØ1Ø1Ø
1ØØ	Ø11ØØ1ØØ

Con el sistema binario puedes escribir cualquier número. Puedes comprobarlo con la siguientes función:

8. BIN$: Convierte un número decimal en su equivalente del sistema binario.

PRINT BIN$ (N)(N es el número que queremos convertir).

Ejemplo:

```
PRINT BIN$ (7328)
    111ØØØ1Ø1ØØØØØ
```

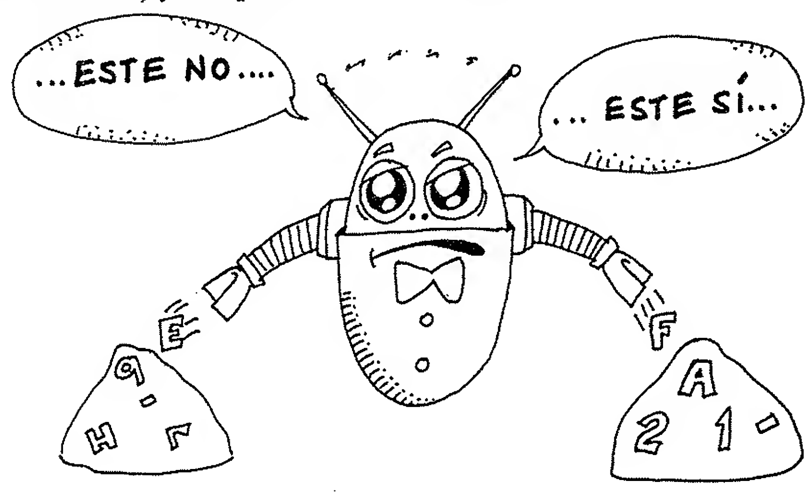


IF THEN

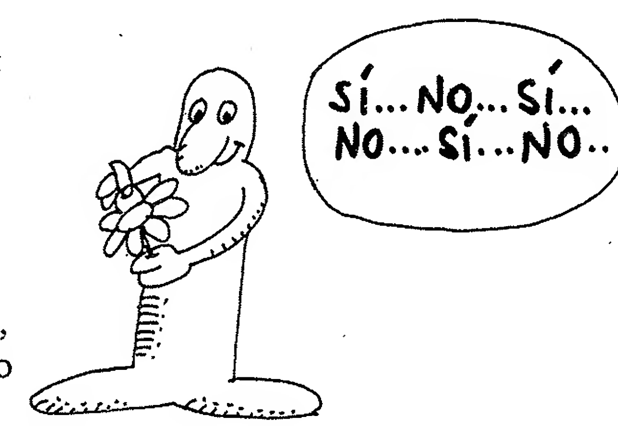
Ahora, vamos a ver lo de hoy: IF THEN

El flujo es de un programa es la dirección y forma en que el programa se desarrolla. El BASIC es un idioma lógico, ordenado. Cada instrucción tiene una misión, sirve para algo concreto. El BASIC es un idioma lógico que basa toda su fuerza, su potencia, en las comparaciones que hace.

Digamos que el ordenador es ante todo una calculadora.
Pues bien, eso es sólo media verdad.
El ordenador es, también ante todo una comparadora:
Una máquina que compara datos, va eliminando los que no le sirven, y se queda con los que sí le sirven.



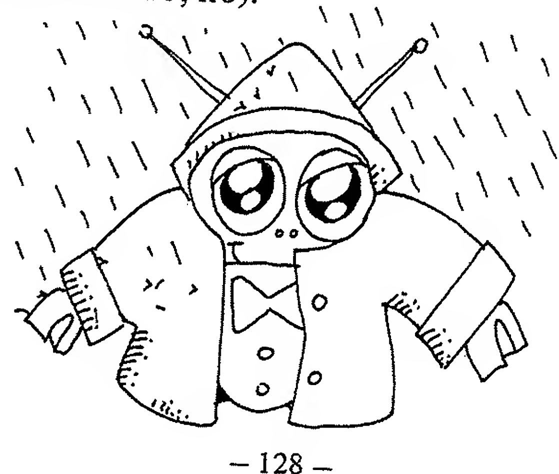
El Basic calcula y compara. (Realmente eso hacen todos los idiomas informáticos y ordenadores: calcular y comparar, y lo hacen de un modo muy simple:
Comprueban
si el dato
es bueno o malo:
sí, no, sí, no:
1, Ø, 1, Ø, 1, Ø.
Eso se llama
Sistema Binario,
que es el
lenguaje máquina,
el lenguaje interno
del ordenador).



El Basic es un comparador. ¿Cuál es la instrucción con que se hacen las comparaciones?

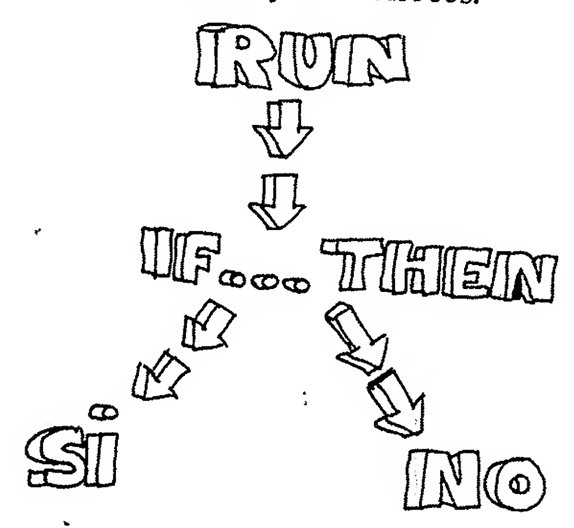
IF. . . THEN. Significa SI. . . ENTONCES

IF está lloviendo THEN me pongo el impermeable. (Si no está lloviendo, no).



Es lógico.

IF oigo el timbre THEN abro la puerta.
(Si no suena, no).
El flujo de IF THEN ya lo conoces.



Pero a partir de IF. . . THEN, dependiendo de si se cumple o no la condición, el flujo podrá adoptar cualquier camino, ya que, después de IF. . . THEN se pone una instrucción, y será ésta la que dirija el flujo.

IF THEN tiene unos íntimos colaboradores: los OPERADORES LOGICOS o SIGNOS DE COMPARACION.

- > Mayor que
- >= Mayor o igual que
- = Igual
- < Menor que
- <= Menor o igual que
- <> Distinto que

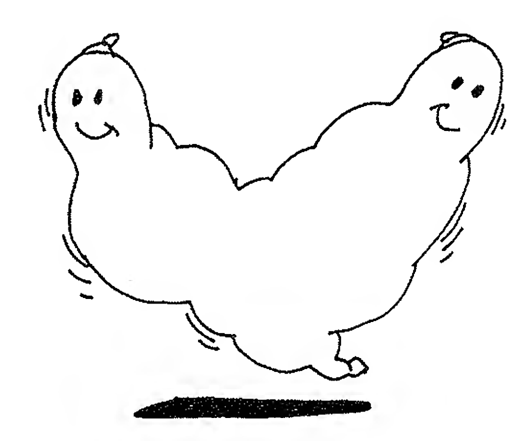
que ya sabes como actúan.

También tiene otros colaboradores interesantes, entre los que destaca:

ELSE

IF oigo pitido THEN pulso la barra espaciadora ELSE no. SI oigo el pitido ENTONCES pulso la barra espaciadora si no no.

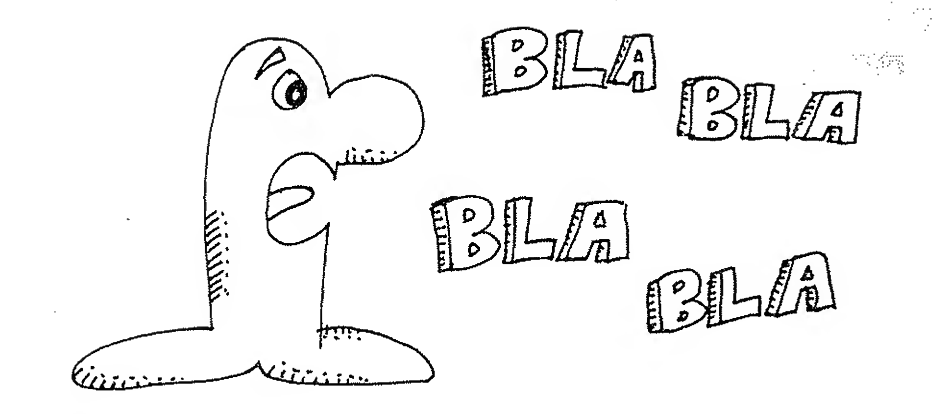
IF. . . THEN. . . es un BIFURCACION



Ahora vamos a "pegarnos" un descansillo.

Cada vez conocemos mejor el idioma Basic. Conoces acciones (instrucciones principales); funciones (instrucciones dependientes de otras); frases.

Sí. Una FRASE, también llamada SENTENCIA, es una instrucción completa, con significado. Es como una oración gramatical, como una frase de cualquier idioma. Una frase es una línea con sentido.



TECLAS DE FUNCION

Ahora vamos a hacer, antes de meternos en faena, una disgresión, para ahorrarte trabajo, ya que cada día tienes que teclear más instrucciones y programas. ¿Recuerdas que en la primera lección te hablamos de las teclas de función?

Hoy vamos a darle un repaso a todas:

F1 - COLOR. Eligeel color de la pantalla.

F2 - AUTO. Te va poniendo automáticamente el número de línea. Se bloquea con **CTRL** **STOP**

F3 - GOTO. Ya sabes.

F4 - LIST. También

F5 - RUN. Por supuesto.

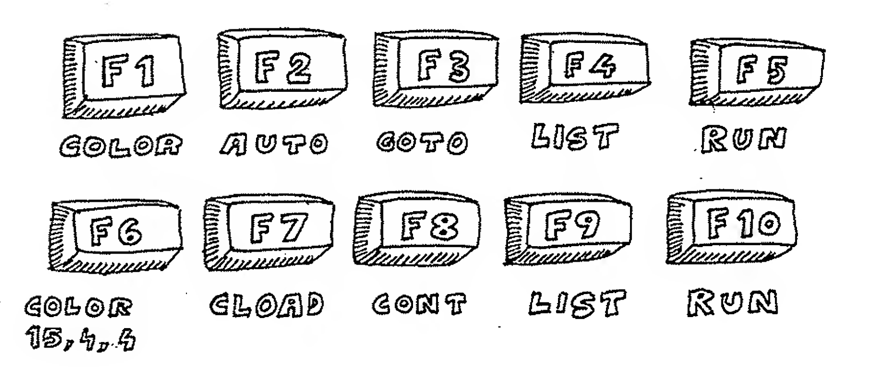
SHIFT F6. COLOR 15, 4, 3. Estés en el color que estés, vuelve al 15, 4, 4 : letras blancas, fondo azul, marco azul (en SCREEN 2 ó 3, paciencia).

SHIFT F7. CLOAD". Conocidísima.

SHIFT F8. CONT. También.

SHIFT F9. LIST. Te lista la última línea que has modificado; aquella en que se ha producido un error, o la última línea del programa recién cargado.

SHIFT F10. RUN. La conoces, pero en este caso tiene doble utilidad, porque antes de ejecutar el programa, hace un **SHIFT** **CLR** (borrado de pantalla).



Las instrucciones contenidas en las teclas de función pueden ser modificadas con la clave KEY.

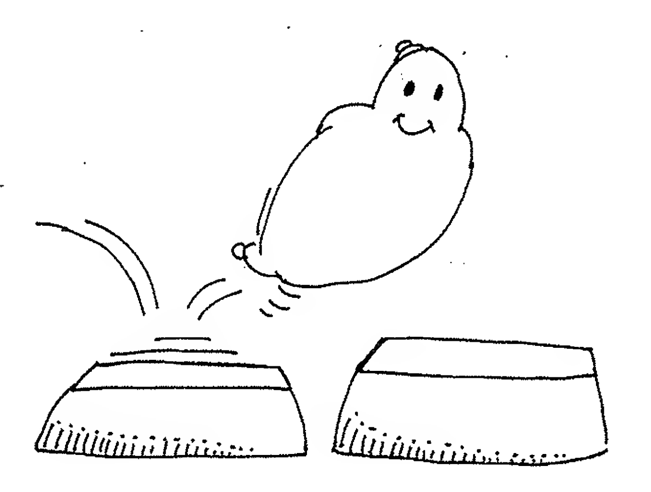
KEY 2, "CSAVE"

(El número es el de la tecla que queremos modificar).

Con KEY OFF limpiamos de la pantalla los mensajes de las teclas de función.

Con KEY ON volvemos a activarlos.

Lo demás corre de tu cuenta. Haz prácticas con los programas de hoy.



PRACTICAS PROGRAMAS TAREAS

```
10 REM EJEMPLO DE MAYOR O IGUAL QUE
20 A=0
30 A=A+2
40 PRINT A
50 IF A>=100 THEN PRINT "FIN":END
60 GOTO 30
```

42

```
10 REM EJEMPLO DE MENOR QUE
20 A=100
30 A=A-2
40 PRINT A
50 IF A<0 THEN PRINT "FIN":END
60 GOTO 30
```

43

```
10 REM EJEMPLO DE DISTINTO
20 INPUT "INTRODUCE UN NOMBRE";A$
30 INPUT "INTRODUCE OTRO NOMBRE";B$
40 IF A$<>B$ THEN PRINT "NO SE TRATA
DE LA MISMA PERSONA:GOTO 60
50 PRINT "SON LA MISMA PERSONA"
60 GOTO 20
```

44

```
10 REM EJEMPLO DE IGUAL QUE
20 REM ESTE PROGRAMA NO PARARA PORQUE
¡A¿ NUNGA VALDRA 100
30 A=1
40 A=A+2
50 PRINT A
60 IF A=100 THEN PRINT "FIN":END
70 GOTO 40
```

45

```
10 REM EJEMPLO DE MENOR O IGUAL QUE
20 A=100
30 A=A-2
40 PRINT A
50 IF A<=0 THEN PRINT "FIN":END
60 GOTO 30
```

46


```
10 REM <<<< LOTERIA PRIMITIVA >>>>
20 X=0
30 FOR I=1 TO 6
40 A=INT(RND(1)*49+1)
50 IF A=X(1) GOTO 40
60 IF A=X(2) GOTO 40
70 IF A=X(3) GOTO 40
80 IF A=X(4) GOTO 40
90 IF A=X(5) GOTO 40
100 X(I)=A
110 NEXT I
120 PRINT "LOS NUMEROS DE LOTERIA SER
AN:"
130 FOR I=1 TO 6:PRINT X(I):NEXT I:PR
INT
140 INPUT "DOY OTRA RONDA (S/N),";A$
150 IF A$="S" GOTO 20
160 PRINT "SUERTE"
170 END
```

47

```
10 REM PINTA
20 COLOR4,4,4:SCREEN2
30 FORY=0TO192
40 LINE(0,Y)-(127,95),8:LINE(127,95)-
(255,Y),8
50 NEXTY
60 GOTO60
```

48

```
10 REM EJEMPLO DE MAYOR QUE
20 A=0
30 A=A+2
40 PRINT A
50 IF A>100 THEN PRINT "FIN":END
60 GOTO 30
```

49

```
10 REM RAYAS DE COLORES
20 SCREEN 2:COLOR 15,7,7:CLS
30 FOR X=1 TO 255 STEP 2
40 LINE(120,90)-(X,0),15
50 LINE(120,90)-(X,192),11
60 LINE(X,0)-(X,192),9
70 NEXT X
80 GOTO 80
```

50

Lección 18

Te estarás luciendo con tus amigos, ¿no?. No es para menos. El que sabe manejar un ordenador puede presumir de ello, porque la informática es una herramienta para personas despiertas, curiosas y aficionadas a la vida.



Hoy vamos a seguir sumando conocimientos a nuestro contador particular de BASIC. Vamos a presentarte la instrucción FOR. . . TO. . . NEXT.

AVANCE REPASO

1.- IF. . . THEN. Es una bifurcación. Esta instrucción establece una comparación, dependiendo del resultado tomará un camino u otro. Su formato es:
IF V = N THEN INSTRUCCION (ELSE INSTRUCCION).

2.- KEY. Con esta clave podemos:

1. Modificar el contenido de las teclas de función.

   KEY N, "Mensaje"

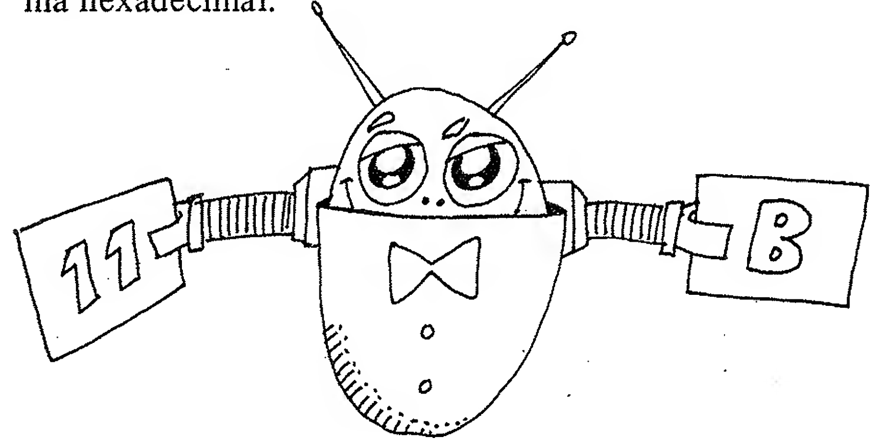
2. Borrar los mensajes de las teclas de función que salen en la franja inferior de la pantalla, o activarlos.

   KEY OFF desactiva

   KEY ON activa

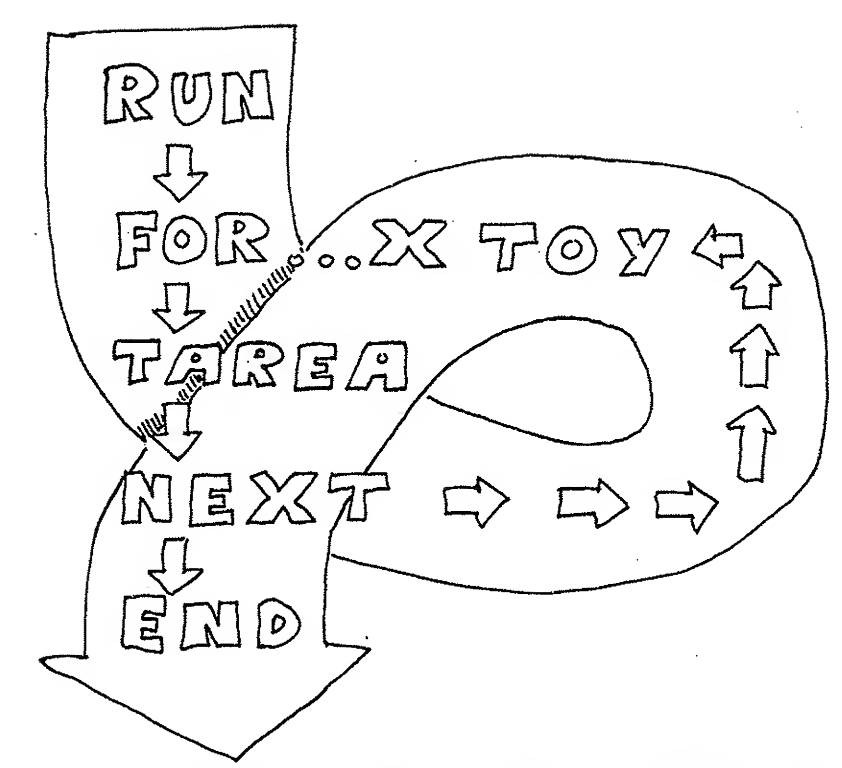
3.- BIN$. Convierte un número del sistema decimal en el equivanlente del sistema binario.

4.- HEX$. Hace otro tanto pero convirtiéndolo al sistema hexadecimal.



Vamos ahora con FOR. . . TO. . . NEXT.

FOR. . . TO. . . NEXT es un bucle controlado. Es un GOTO con salida, un GOTO con fin. FOR. . .NEXT hace que se repita una tarea un número de veces determinado. Ya conoces su flujo:



Esa es la principal misión de FOR . . . NEXT: repetir una tarea varias veces. Pero ¿Y si no se le pone tarea qué hace?



TIME

Pero también existe en BASIC otro controlador de tiempo, otro temporizador, que es todo un reloj: Se trata de TIME. TIME es una variable especial que sirve como controlador de tiempo. Es como una función que actúa asociada a PRINT.

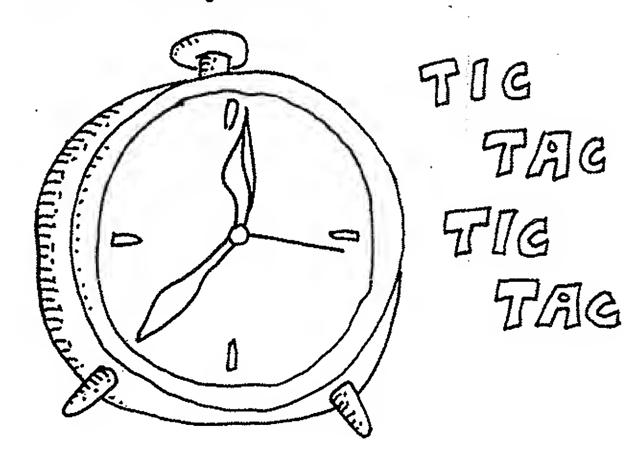
TIME te dice la hora, el tiempo transcurrido desde que enciendes tu ordenador, hasta el momento en que te hallas. TIME funciona asociado con IF THEN.

```
1Ø  PRINT "EMPIEZA EL TIEMPO"
2Ø  TIME = Ø
3Ø  IF TIME/5Ø = 3Ø THEN 4Ø
35  GOTO 3Ø
4Ø  PRINT "SE ACABO EL TIEMPO"
5Ø  END
```

Su estructura es muy simple.

```
TIME = Ø
IF TIME/5Ø = T THEN INSTRUCCIONES
          (T = tiempo en segundos)
GOTO L
          (L = línea del IF TIME/5Ø)
```


TIME es un reloj exacto.



Continuemos con FOR. . . NEXT.

FOR. . .NEXT tiene un complemento : STEP.

STEP sirve para variar el ritmo del contador FOR... NEXT, haciendo que éste cuente con el intervalo o paso que nosotros deseemos.

```
1Ø  FOR  T1 = Ø TO 222  STEP 1Ø
2Ø  PRINT T1
3Ø  NEXT T1
```

Ejecútalo.

Como verás, cuenta de mil, en mil, y muy rápido. No da tiempo a ver los números bien. ¿No podríamos hacer que fuese más despacio? Sí podemos, pues claro que podemos. En BASIC se puede hacer todo. El ordenador hará todo lo que tu seas capaz de enseñarle a hacer.

¿Y cómo lo hacemos?

Pues intercalando otro TEMPORIZADOR que retarde la ejecución del primero.

Introduce la línea 15 en el programa anterior.

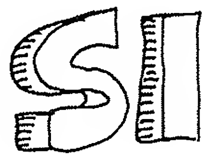
15 FOR T2 = 1 TO 1ØØØ : NEXT T2

¿Qué te ha parecido.?

FOR. . . NEXT es un buen temporizador.

Tengo que decirte algo : cuando utilices más de un FOR. . . NEXT en un programa, tienes que evitar que sus bucles se crucen.

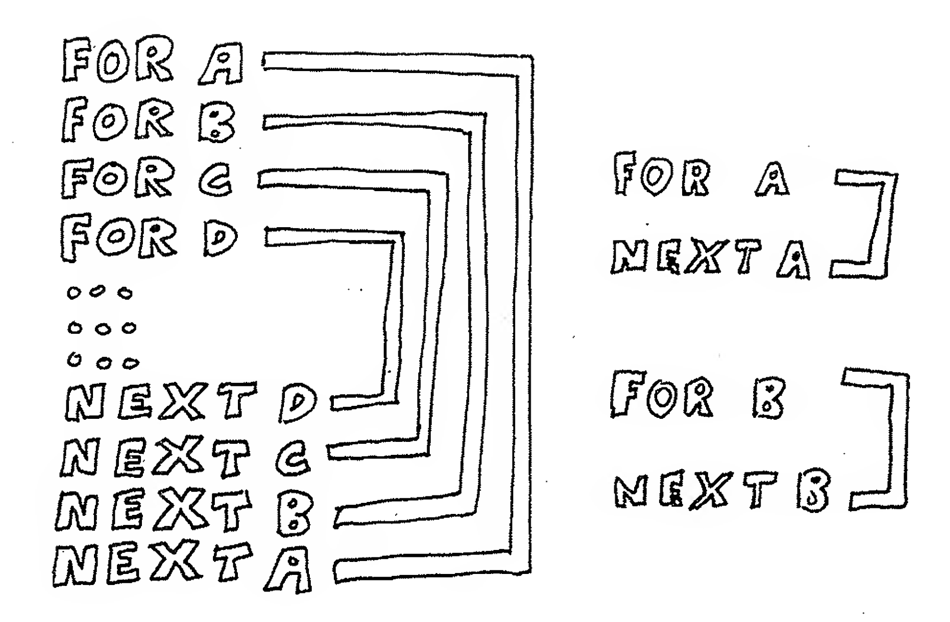
```
1Ø  FOR X TO
2Ø  FOR Y TO
3Ø  NEXT Y
4Ø  NEXT X
```




```
1Ø  FOR X
2Ø  FOR Y
3Ø  NEXT X
4Ø  NEXT Y
```



Cuando hay más de un bucle FOR . . . NEXT en un programa, tienen que quedar perfectamente encajados, de modo que sus flujos no se crucen.O quedar separados.



Y una última cosa que he de decirte: si se te olvida poner en el programa el FOR, o cruzas dos bucles, cometerás un error y el ordenador te lo advertirá con el mensaje:

NEXT WITHOUT FOR

Bueno, y como el movimiento se demuestra andando, ahí te dejo las prácticas de cada día.

PRACTICAS PROGRAMAS TAREAS

```
10 REM ESTE PROGRAMA TE OBTIENE
CUALQUIER TABLA DE SUMAR
20 CLS
30 INPUT "TABLA DEL NUMERO";N
40 FOR M=1 TO 10
50 PRINT N;"+";M;"=";N+M
60 NEXT M
70 GOTO 30
```

51

```
10 REM ESTE PROGRAMA TE OBTIENE
CUALQUIER TABLA DE RESTAR
20 CLS
30 INPUT "TABLA DEL NUMERO";N
40 FOR M=1 TO 10
50 PRINT N;"-";M;"=";N-M
60 NEXT M
70 GOTO 30
```

52

```
10 REM ESTE PROGRAMA TE OBTIENE
CUALQUIER TABLA DE MULTIPLICAR
20 CLS
30 INPUT "TABLA DEL NUMERO";N
40 FOR M=1 TO 10
50 PRINT N;"*";M;"=";N*M
60 NEXT M
70 GOTO 30
```

53

```
10 REM ESTE PROGRAMA TE OBTIENE
CUALQUIER TABLA DE DIVIDIR
20 CLS
30 INPUT "TABLA DEL NUMERO";N
40 FOR M=1 TO 10
50 PRINT N;"/";M;"=";N/M
60 NEXT M
70 GOTO 30
```

54

55

```
10 REM GRAFICAS DE FUNCIONES
15 CLS
40 PRINT
50 FOR Y=6 TO -6 STEP -1
60 PRINTY;
70 FOR X=-10 TO 10
80 IF (SQR(ABS(Y*X*2))-X)>.5 THEN PRI
NT"X"; ELSE PRINT".";
90 NEXT X
100 PRINT
110 NEXT Y
120 PRINT"    098765432101234567890"
130 REM CAMBIAR LA FUNCION QUE HAY
ENTRE LOS PRIMEROS PARENTESIS DE LA
LINEA 80 PARA OBTENER LA REPRESENTACI
ON DE ESTA.
```


56

```
10 REM RELOJ DESPERTADOR
20 COLOR1,15,15:SCREEN1
30 INPUT "HORAS:     ";H1
40 INPUT "MINUTOS:   ";M1
50 INPUT "SEGUNDOS:  ";S1
60 CLS
70 TIME=0
80 T=TIME
90 H=INT(T/180000!)
100 T=T-(H*180000!)
110 M=INT(T/3000)
120 T=T-(M*3000)
130 S=INT(T/50)
140 IF H1=H AND M1=M AND S1=S THEN PL
AY"V15O5C06D"
150 LOCATE0,0
160 PRINTUSING "&&:&&:&&";H;M;S
170 GOTO80
```

57

```
10 REM MOVER POR PANTALLA
20 CLS:INPUT "INTRODUCE X,Y";X,Y
30 SCREEN 2:COLOR 15,4,4:CLS
40 B$=INKEY$:IF B$>CHR$(27) AND
   B$<CHR$(32) THEN A$=B$
50 IF A$=CHR$(28) AND X<240 THEN
   X=X+1
60 IF A$=CHR$(29) AND X>10 THEN X=X-1
70 IF A$=CHR$(30) AND Y>10 THEN Y=Y-1
80 IF A$=CHR$(31) AND Y<180 THEN
   Y=Y+1
90 PSET(X,Y),15
100 GOTO 40
```

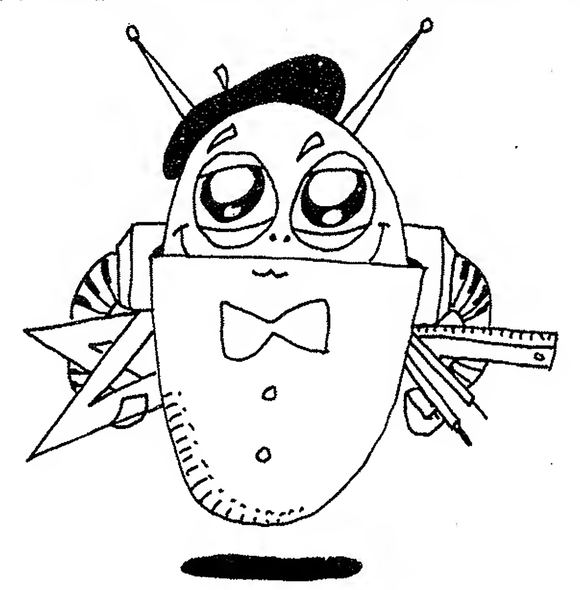
```
10 REM EJEMPLO DE LOCATE
20 SCREEN 0:COLOR 15,4,4:CLS
30 FOR X=0 TO 31 STEP 6
40 FOR T=1 TO 300:NEXT T
50 LOCATE X,10:PRINT "LOCATE"
60 NEXT X
70 FOR Y=0 TO 24
80 FOR R=1 TO 300:NEXT R
90 LOCATE 14,Y:PRINT "LOCATE"
100 NEXT Y
110 END
```

58

Lección 20

Se nos ha terminado la lógica. Quiero decir que ya hemos visto, y espero que conozcas, las más importantes instrucciones que desarrollan los cálculos y comparaciones de tu MSX Naturalmente esto no son sino los primeros balbuceos de un idioma.

Ya podemos decir que sabemos cómo piensa tu ordenador y sabemos qué palabras utilizar para comunicarnos con él en BASIC. Ya sabemos cómo escribe, cómo piensa, cómo cálcula. Ahora vamos a ver cómo dibuja. Pero antes, como a diario, vamos a curiosear en lo que sabemos, y un poco en lo que nos queda por saber.



AVANCE REPASO

1.- FOR NEXT. Es un bucle controlado, un controlador, un contador, un temporizador.

2.- NEX WITHOUT FOR. Es el error que se produce cuando se pone incompleta la instrucción FOR NEXT. También se produce cuando se cruzan los flujos de los bucles.

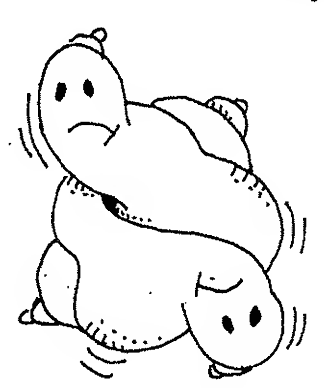
3.- TIME. Es una variable que actúa unida a IF THEN. Funciona como un reloj, y tiene el siguiente formato.

TIME = 0

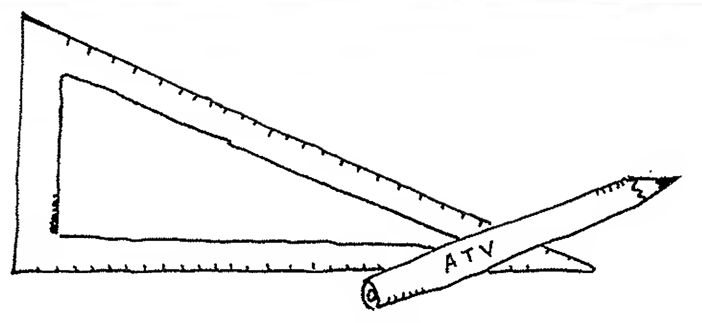
IF TIME/50 OPERADOR LOGICO Nº DE SEGUNDOS THEN INSTRUCCION

GOTO LINEA DEL IF TIME/50...

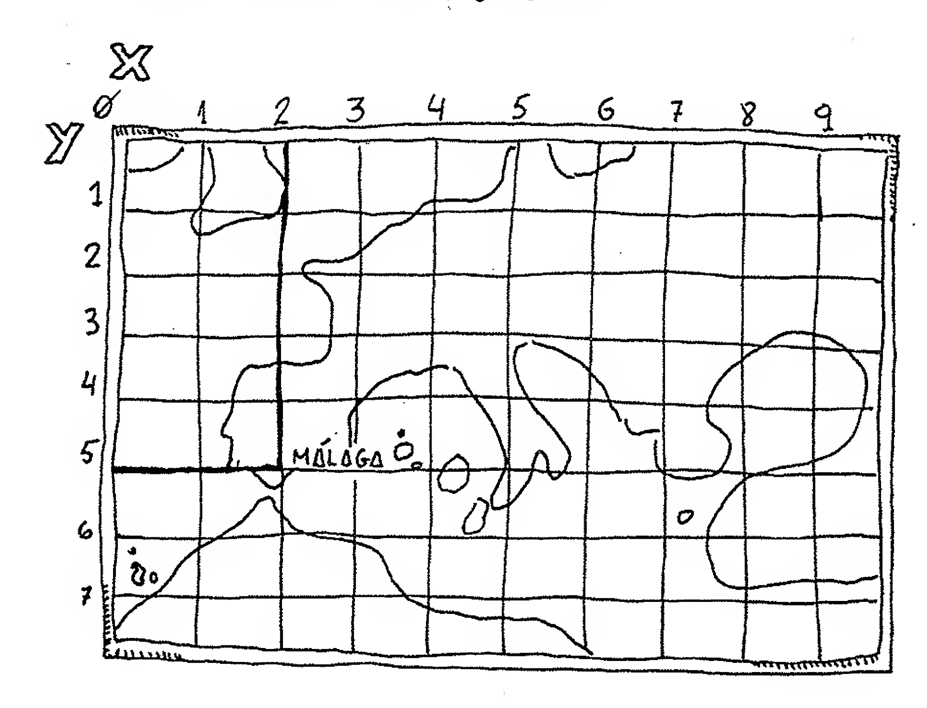
4.- BUCLES ENCAJADOS O ANIDADOS. En un programa se pueden poner varios bucles FOR NEXT, pero evitando que se crucen sus flujos.



VAMOS AL GRANO

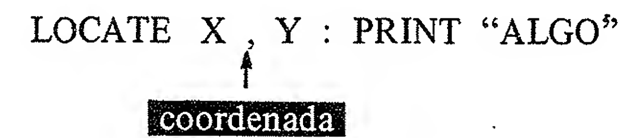


Y vamos a dibujar. Esta lección requiere pocas explicaciones, porque ya hemos visto todas las instrucciones en la lección 19. No obstante, ¿Tú sabes qué es una coordenada? ¿Estás seguro? ¿Seguro?

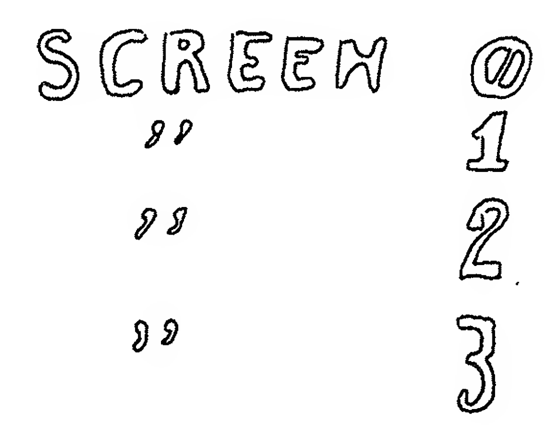


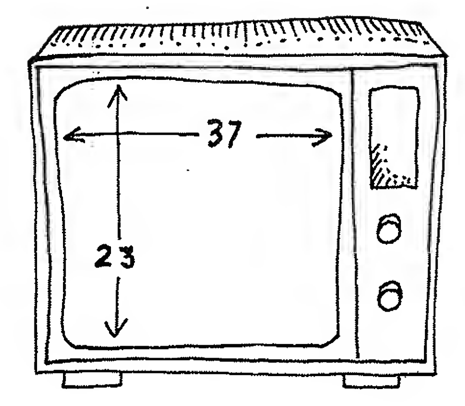
Málaga está en ese mapa, en la coordenada (2,5). Es decir, está a 2 cuadritos de la izquierda, y a 5 de arriba.

LOCATE significa "situar en". LOCATE, como suena. Es una instrucción que funciona asociada a PRINT. Su formato:



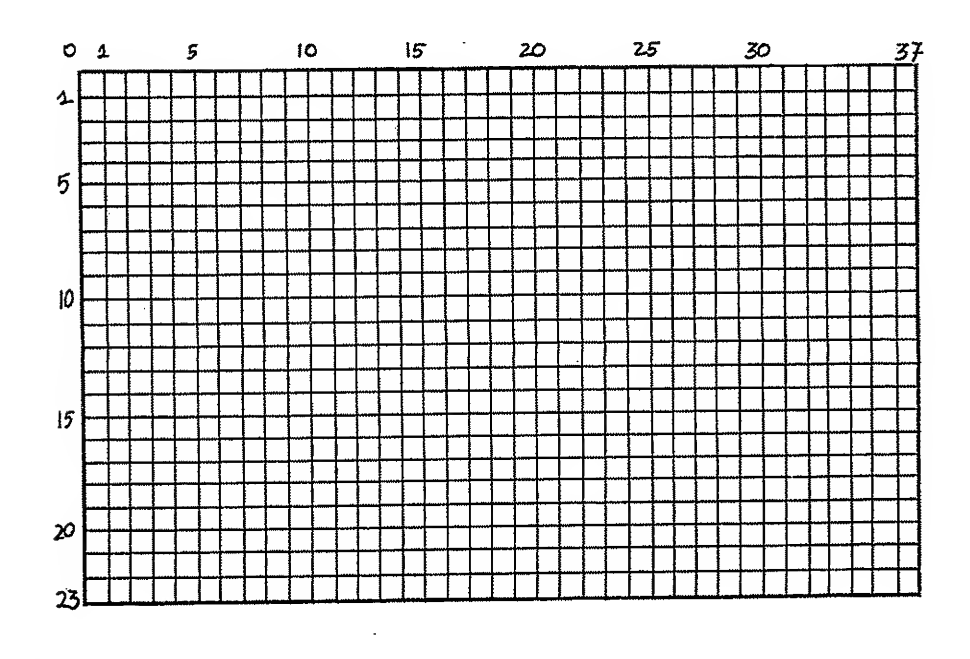
Así es como funciona LOCATE: situa una "cadena", utilizando PRINT, en donde le digamos. Tenemos que decirle las coordenadas X e Y. Pero ¿de qué pantalla?. En la lección 19 vimos varias instrucciones gráficas en SCREEN 2, que también se pueden utilizar en SCREEN 3. Pero también tenemos en nuestro MSX SCREEN Ø y SCREEN 1 . Los tamaños de las pantallas de SCREEN Ø y SCREEN 1, que son los que utiliza LOCATE: PRINT son:

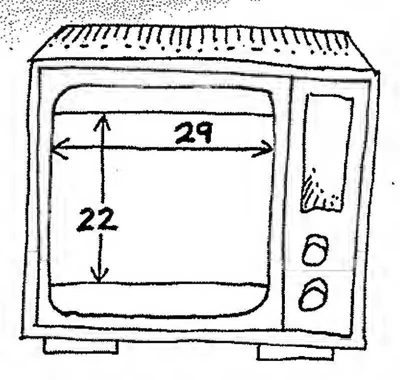




SCREEN Ø

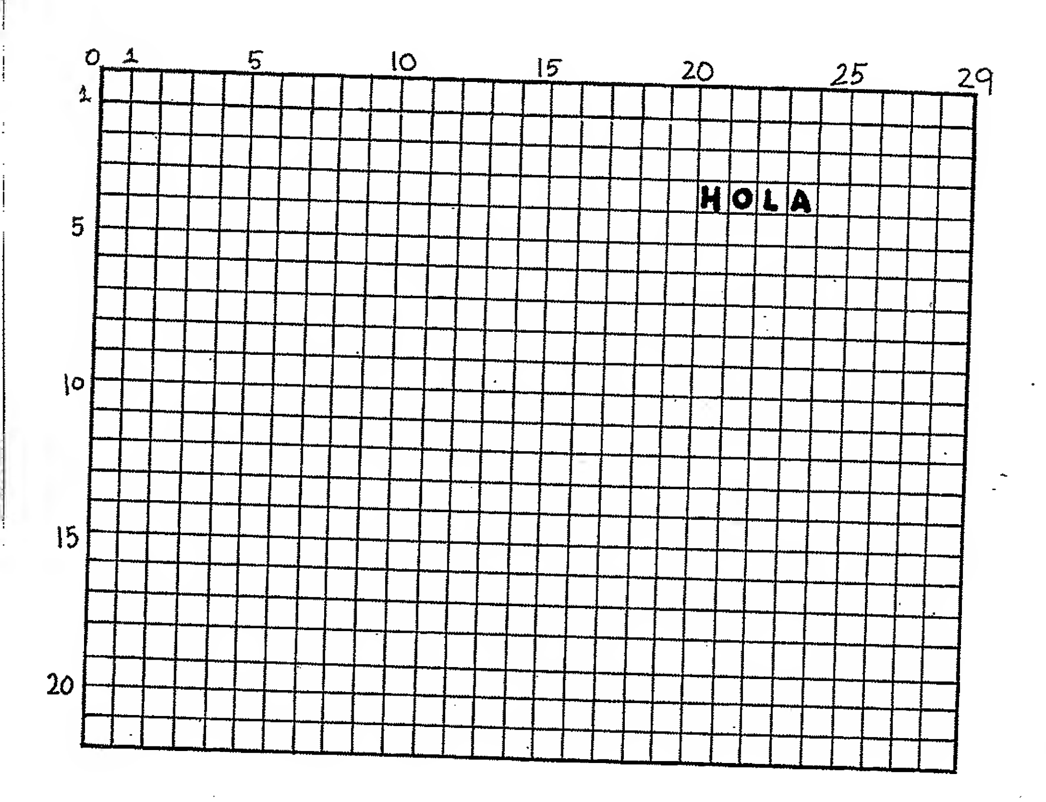
EN ESTA PANTALLA CABEN 851 CARACTERES





SCREEN 1

EN ESTA PANTALLA CABEN 638 CARACTERES




```
10 CLS
20 SCREEN 1
30 LOCATE 14,11 : PRINT "HOLA"
40 GOTO 40
```

Con este programa decimos:

10 Limpia la pantalla
20 Quiero escribir en SCREEN 1
30 Situa en la posición 14,11 el texto PRINT : HOLA
40 Bloquea el programa para que permanezca el texto en la pantalla.

Bien, pues eso es LOCATE : PRINT.

Y ahora, vamos a repasar un poco el formato de las instrucciones gráficas que vienen en la lección.

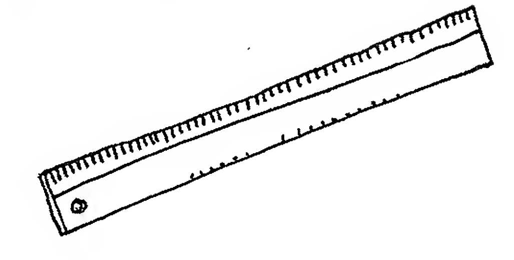
PSET (Coordenada X, coordenada Y), Color



LINE (CX , CY) – (CX , CY) , Color

coordenada del punto de inicio	coordenada del punto de fin

LA REGLA



LINE tiene dos subcomandos de interés.

B y **BF**

Ejecuta ese programa.

```
1Ø  CLS
2Ø  SCREEN 3
3Ø  LINE (1ØØ, 7Ø) – (15Ø, 14Ø), 8
4Ø  GOTO 4Ø
```

SHIFT **F10**

Ahora, añade esto al final de la línea 3Ø:

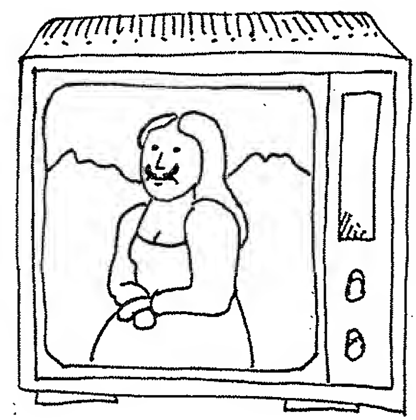
, **B** , y ejecútalo.

Y ahora añade al final de la misma línea
F

,B al final de LINE hace un cuadrado.

,BF al final de LINE hace un cuadrado y lo llena de color.

El ordenador dibujará los dibujos que tú seas capaz de dibujar



CIRCLE (Coordenada X , Coordenada Y), Radio, , , , , 1.4

Punto central del círculo	siempre igual

Si queremos elegir un color para el trazo del círculo, hay que añadir después de RADIO, una coma **,** ; poner el color deseado, y luego, de las cuatro comas, borrar una.

CIRCLE (X , Y), Radio, Color , , , 1.4

Por hoy está bien. Ahora dedica todo el tiempo que quieras a las prácticas.

PRACTICAS PROGRAMAS TAREAS

59

```
10 REM ROMPECABEZAS
15 H=RND(-TIME)
20 DIM B(4,4)
30 REM INICIO
40 FOR R=1 TO 4
50 FOR C=1 TO 4
60 FG=INT(RND(1)*3)
65 IF FG=0 THEN GOTO 60
66 B(R,C)=FG
70 NEXT C,R
80 CLS
100 REM DIBUJO DE PANTALLA
110 LOCATE15,5:PRINT"1234"
120 FOR R=1 TO 4
130 PRINT:PRINT TAB(12);R;
140 FOR C=1 TO 4
160 PRINT MID$("AB",B(R,C),1);
170 NEXT C,R
200 REM PONER MOVIMIENTO
210 PRINT:PRINT
220 INPUT "FILA=";R
230 IF R<1 OR R>4 THEN 220
240 FOR C=1 TO 4
250 B(C,R)=3-B(R,C)
260 NEXT C
270 INPUT "COLUMNA=";C
280 IF C<1 OR C>4 THEN 270
290 B(R,C)=3-B(R,C)
300 FOR R=1 TO 4
310 B(R,C)=3-B(R,C)
320 NEXT R
330 GOTO 100
340 REM TRATA DE FORMAR EL ROMPECABE
ZAS, Y QUE SE FORME ESTA FIGURA:
                 ABBA
                 ABBA
350 REM          ABBA
                 ABBA
```

60

```
10 REM MOVIMIENTOS
20 SCREEN 2:COLOR 15,4,4:CLS
30 Y=30:Y1=190:PI=3.1416
40 PSET(190,0),11
50 FOR T=0 TO 6*PI STEP PI/25
60 XX=X:X=60*COS(T)+126
70 YY=Y:Y=Y+1:YB=Y1:Y1=Y1-1
80 LINE(120,0)-(X,Y1),11
90 LINE(126,Y)-(X,Y1),11
100 LINE(120,0)-(XX,YB),4
110 LINE(126,YY)-(XX,YB),4
120 NEXT T
130 GOTO 130
```

61

```
10 REM FIGURA DE COSENO
20 SCREEN 2:COLOR 15,4,4:CLS
30 Y=30:Y1=190:PI=3.1416
40 FOR T=0 TO 6*PI STEP PI/25
50 X=60*COS(T)+126
60 Y=Y+1:Y1=Y1-1
70 LINE(X,Y)-(120,Y1),11
80 NEXT T
90 GOTO 90
```

62

```
10 REM <<< EJEMPLO DE LINE >>>
20 SCREEN2:COLOR15,1,1:CLS
30 LINE(20,0)-(71,20),11,BF
40 LINE(20,28)-(35,70),11,BF
50 LINE(43,28)-(71,70),11,BF
60 LINE(79,0)-(245,20),10,BF
70 LINE(79,28)-(180,70),4,BF
80 LINE(188,28)-(245,120),15,BF
90 LINE(188,128)-(245,150),15,BF
100 LINE(130,78)-(180,150),6,BF
110 LINE(43,78)-(122,150),6,BF
120 LINE(20,78)-(35,180),13,BF
130 LINE(43,158)-(122,180),15,BF
140 LINE(20,188)-(122,194),10,BF
150 LINE(130,158)-(180,194),13,BF
160 LINE(188,158)-(245,194),4,BF
170 GOTO 170
```

63

```
10 REM //// cuadro ///
20 SCREEN3:COLOR 1,14,14:CLS:CT=1:X1=
0:Y1=0:X2=0:Y2=0
30 CT=CT+1:IF CT>13 THEN CT=2
40 LINE (128+X2,95-(Y2+Y1))-
(128+(X2+X1),95-Y2),CT
50 LINE (128+(X2+X1),96+Y2)-
(128+X2,96+(Y2+Y1)),CT
60 LINE (127-X2,95-(Y2+Y1))-
(127-(X2+X1),95-Y2),CT
70 LINE (127-(X2+X1),96+Y2)-
(127-X2,96+(Y2+Y1)),CT
80 IF Y1<92 THEN Y1=Y1+4:X1=X1+4 ELSE
X2=X2+4
90 IF X1+X2>127 THEN Y2=Y2+4
100 IF Y1<95 AND Y2<95 THEN 30
110 GOTO110
```

64

```
10 REM CUADRO.
20 SCREEN3:COLOR15,1,1:CLS
30 LINE(20,0)-(71,20),11,BF
40 LINE(20,28)-(35,70),11,BF
50 LINE(43,28)-(71,70),11,BF
60 LINE(79,0)-(245,20),10,BF
70 LINE(79,28)-(180,70),4,BF
80 LINE(188,28)-(245,120),15,BF
90 LINE(188,128)-(245,150),15,BF
100 LINE(130,78)-(180,150),6,BF
110 LINE(43,78)-(122,150),6,BF
120 LINE(20,78)-(35,180),13,BF
130 LINE(43,158)-(122,180),15,BF
140 LINE(20,188)-(122,194),10,BF
150 LINE(130,158)-(180,194),.13,BF
160 LINE(188,158)-(245,194),4,BF
170 GOTO170
```

65

```
10 REM PRINCIPIO
20 COLOR15,15,15:SCREEN3
30 LINE(0,0)-(128,96),12,BF
40 LINE(0,100)-(128,191),10,BF
50 LINE(132,0)-(255,96),6,BF
60 LINE(132,100)-(255,191),4,BF
70 REM CONTROL
80 C%=5*RND(1)
90 IF C%=1 THEN LINE(0,0)-(128,96),3,
BF:LINE(0,0)-(128,96),12,BF
100 IF C%=2 THEN LINE(0,100)-(128,191
),11,BF:LINE(0,100)-(128,191),10,BF
110 IF C%=3 THEN LINE(132,0)-(255,96)
,9,BF:LINE(132,0)-(255,96),6,BF
120 IFC%=4THENLINE(132,100)-(255,191)
,5,BF:LINE(132,100)-(255,191),4,BF
130 GOTO80
```

```
10 SCREEN 2:COLOR 15,1,1:CLS
20 B=20
30 FOR A=1 TO 360 STEP .2
40 XA=X:X=INT(COS(A)*B+COS(A)*B)+120
50 YA=Y:Y=INT((SIN(A)*B+SIN(A)*B)*1.4
   )+60
55 XB=X1:X1=INT(COS(A+A)*B+COS(A+45)*
   B)+120
56 YB=Y1:Y1=INT((SIN(A+A)*B+SIN(A+45)
   *B)*1.4)+80
60 PSET(X,Y),15
65 PSET(X1,Y1),15
70 NEXT A
80 GOTO 80
```

66


```
10 REM CURVA DE SENO
20 SCREEN 2:COLOR 15,4,4:CLS
30 FOR T=0 TO 65 STEP .25
40 X=X+1
50 A=INT(16+15*SIN(T))+90
60 PSET(X,A)
70 NEXT T
80 GOTO 80
```

67

```
10 REM ESPIRALES
20 SCREEN 2
30 COLOR 15,4,4
40 CLS
60 R=2
70 FOR Z%=1 TO 500
80 R=R+.15
90 PSET(128+R*COS(Z%/32*3.142),96+R*S
IN(Z%/32*3.142)),15
100 NEXT Z%
110 R=2
120 FOR Z%=1 TO 500
130 R=R+.15
140 PRESET(128+R*COS(Z%/32*3.142),96+
R*SIN(Z%/32*3.142))
150 NEXT Z%
160 PLAY"L24CDEFEDC"
180 GOTO 10
```

68

69

```
10 REM PINCEL
20 REM PULSA LA 'P' PARA PINTAR
30 REM PULSA LA 'C' PARA CAMBIAR EL C
OLOR DEL PINCEL
40 SCREEN 2:COLOR 15,4,4:CLS:C=15
50 OPEN"GRP:" AS1
60 REM ACEPTA EL TECLADO
70 A$=INKEY$
80 IF A$=CHR$(&H1C) AND X<255 THEN X=
X+1:GOTO 160
90 IF A$=CHR$(&H1D) AND X>0 THEN X=X-
1:GOTO 160
100 IF A$=CHR$(&H1E) AND Y>0 THEN Y=Y
-1:GOTO 160
110 IF A$=CHR$(&H1F) AND Y<170 THEN Y
=Y+1:GOTO 160
120 IF A$="C" THEN GOTO 190
130 PSET(224,0),15:PRINT#1,C;
```

```
140 GOTO 70
150 REM AQUI PINTA
160 PSET(X,Y),C
170 GOTO 70
180 REM AQUI CAMBIA EL COLOR
190 C=C+1:IF C>15 THEN C=0
200 LINE(224,0)-(250,10),0,BF
210 PSET(224,0),15:PRINT#1,C;
220 GOTO 70
```

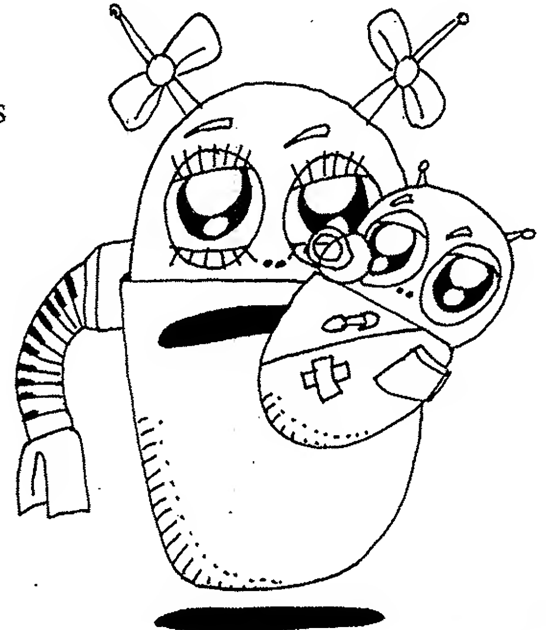
```
10 R=RND(-TIME)
20 REM < LA QUINIELA DE LA SEMANA >
30 FOR Q=1 TO 14
40 N=INT(RND(1)*3+1)
50 IF N=3 THEN PRINT " X":GOTO70
60 PRINT N
70 NEXT Q
80 PRINT"¿OTRA COMBINACION?(S/N)"
90 INPUT R$
100 IF R$="S" THEN GOTO 30
110 PRINT"SUERTE":END
```

70

Lección 22

¡Quién no lo intenta jamás lo conseguirá!

Este es el consejo que introdujo mamá bitilla en la memoria ROM al Profesor Bitillo, cuando era un pequeño amasijo
de chips y tornillos,
con su memoria
RAM
totalmente vacía.
Tú también tienes
que intentarlo.
El único modo
es haciendo
programas.
No me cansaré
de repetírtelo.



```
1Ø PRINT "TIENES QUE HACER PRACTICAS"
2Ø PRINT "POR TU CUENTA"
3Ø GOTO 1Ø
```



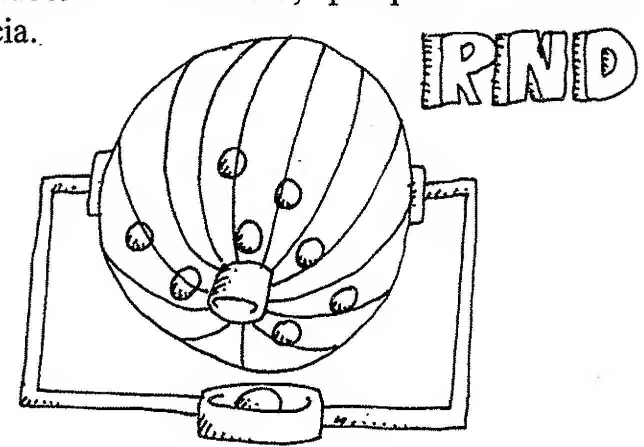
AVANCE REPASO

1. RND. Es una función. Elige un número al azar entre Ø y 1, o entre 1 y el número que tú le digas. Es una función muy útil, aunque un tanto compleja. Pero mejor que le prestes atención porque en los juegos que confecciones no podrás prescindir de ella.

```
1Ø  V = RND ( – TIME)
2Ø  A% = RND (1) x 1ØØ (N = un número)
3Ø  PRINT A%
```

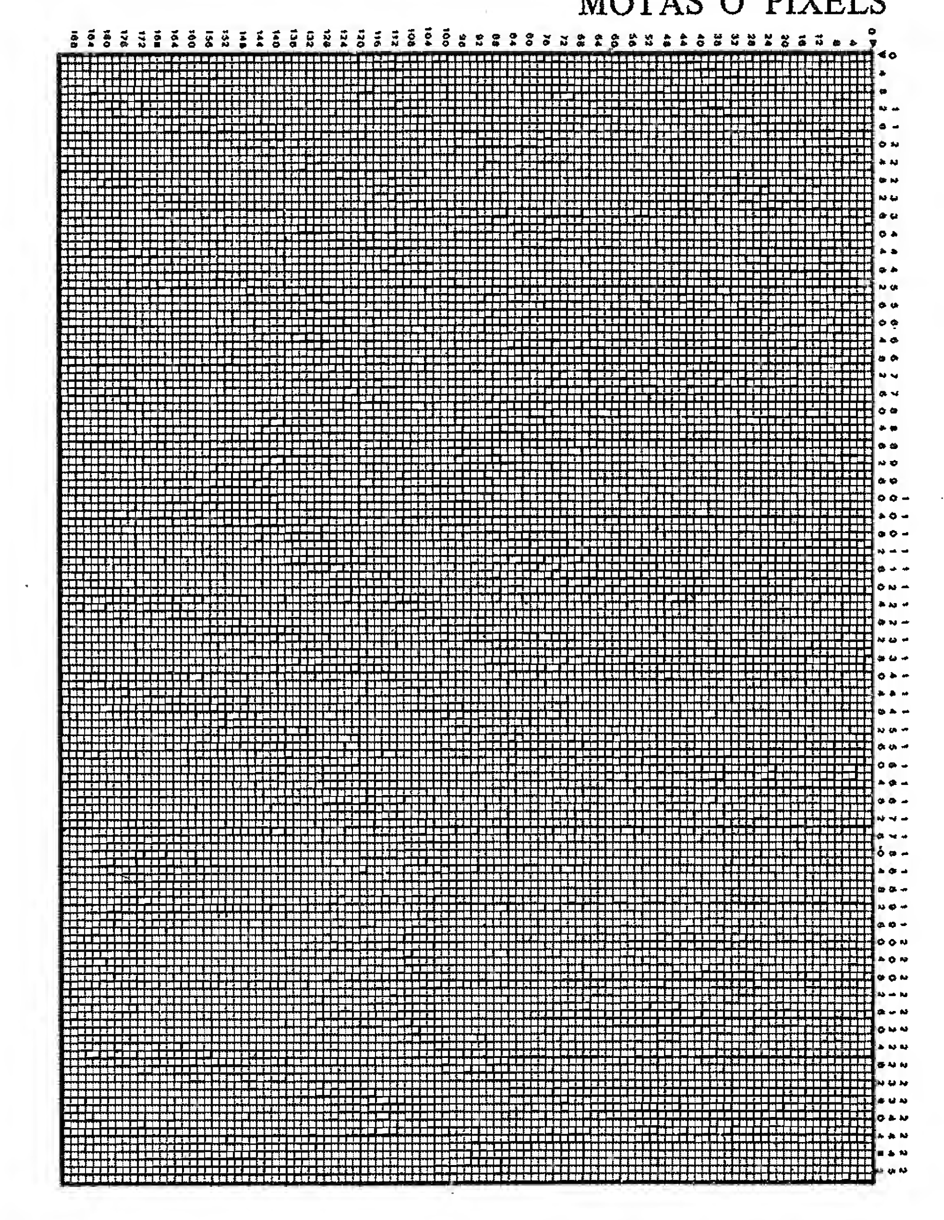
Ejecútalo varias veces. A% cada vez tiene un valor distinto.

Apréndete ese formato, porque lo utilizarás con frecuencia.

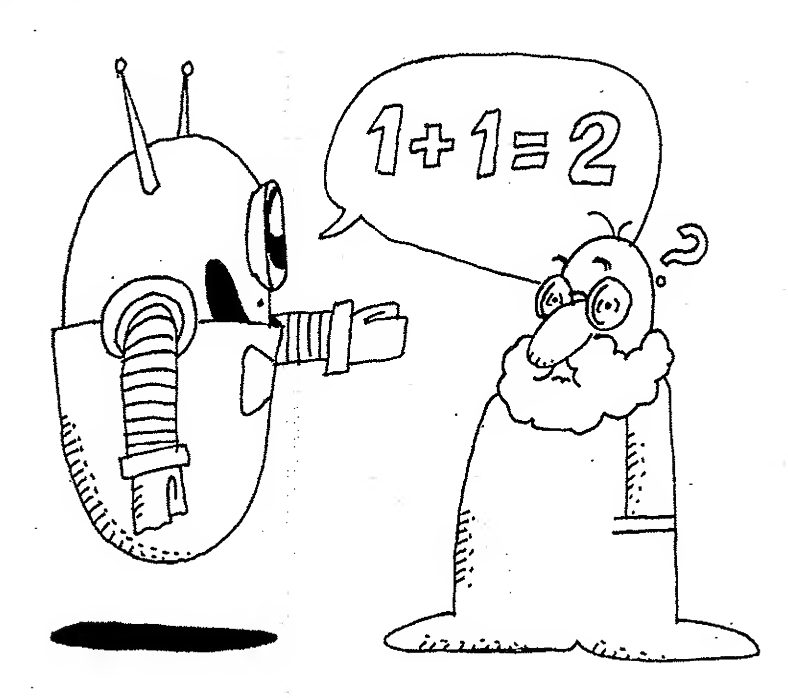


En la lección anterior vimos LOCATE, el SCREEN Ø, el 1, los SCREEN 2 y 3, y algunas instrucciones gráficas que operan en ellos. Ese es el tamaño de los SCREEN 2 y 3 en que se mueven las instrucciones gráficas.

SCREEN 2 256 * 192 = 49152 MOTAS o PIXELS
SCREEN 3 (256/4) * (192/4) = 64 * 48 = 3072 MOTAS O PIXELS



La instrucción pintora del BASIC es DRAW. Ya la conoces. Es una instrucción que permite irla conociendo poco a poco, pero que se tarda tiempo en conocer. Te repito lo que hemos dicho en otras ocasiones : no pretendas llegar a ser un programador en un solo día, porque los programadores con muchos años de experiencia reconocen humildemente que sólo están empezando a aprender.



A DRAW hay que decírselo todo, desde el punto donde ha de empezar a dibujar;

BMX, Y

el color de línea que trazamos; el tamaño del dibujo (escala); y la dirección de la línea.

BMX,Y Cn Sn TRAZOS

Subcomandos de DRAW

M: Desplaza el pincel DRAW por la pantalla. Es decir, con este subcomando le decimos el punto en que queremos empezar a dibujar.

B: Evita que se vea el desplazamiento de M. B pinta invisible, por eso :

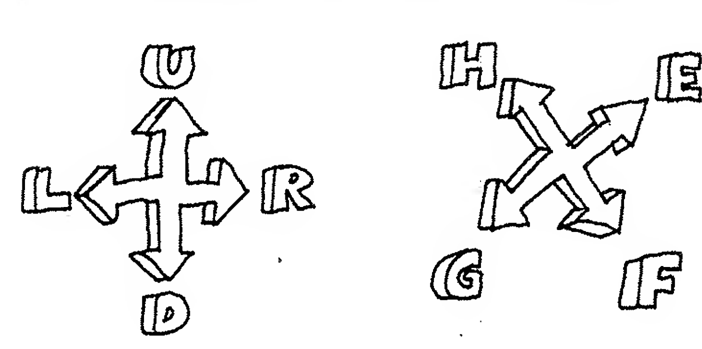
BM: Es el subcomando que utilizaremos para fijar el punto de inicio de un dibujo.

S: Es la escala, el tamaño. Va de 1 a 256.

C: Es el color: Ø a 15.

N: Traza la línea y luego hace que retorne al punto original.

DIRECCIONES DE DRAW



Con eso tienes bastante para empezar.

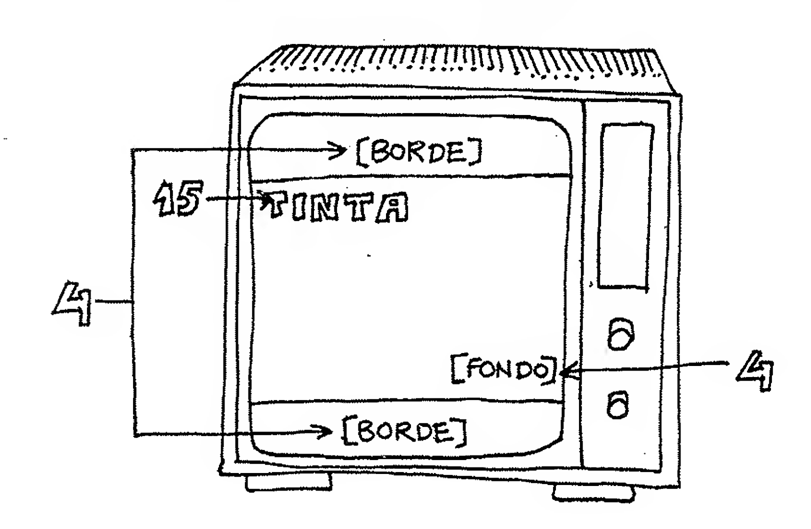
Sí vamos a ver, no obstante, una instrucción que se nos ha quedado en el tintero.

Estamos hablando mucho de pintura. Tenemos lápices, regla, compás, pincel, pero lo que no tenemos es pintura, tinta. La pintura o tinta del Basic es doble : por un lado COLOR, y por otro PAINT.

COLOR sirve para preparar los SCREEN. El COLOR base es el que se obtiene pulsando SHIFT F6 en SCREEN 1.

COLOR 15,4,4 (en algunos modelos de MSX es 15,4,7).

El primer color es el de la TINTA de escritura; El segundo es el color del FONDO de escritura; y el tercero el de los BORDES superior e inferior de la pantalla, en los que no se puede escribir ni dibujar.

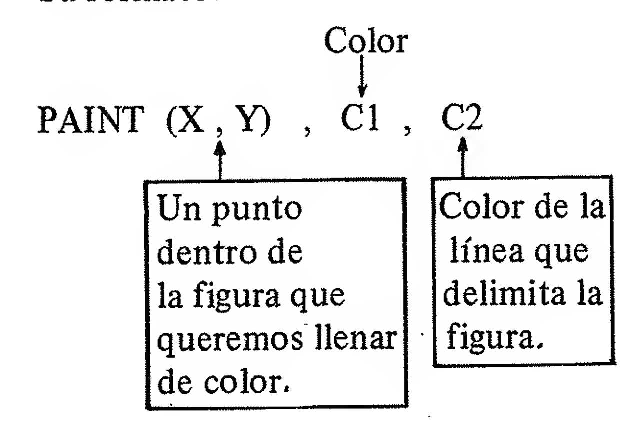


La otra instrucción de que te hablo, la auténtica pintura que sirve para llenar de color los dibujos es PAINT.

Ya sabes que disponemos en tu MSX de 16 colores:

Ø Transparente	
1 = Negro	9 = Rojo pálido
2 = Verde	10 = Amarillo ocre
3 = Verde pálido	11 = Amarillo pálido
4 = Azul oscuro	12 = Verde oscuro
5 = Azul pálido	13 = Magenta
6 = Rojo oscuro	14 = Gris
7 = Ciano	15 = Blanco
8 = Rojo	

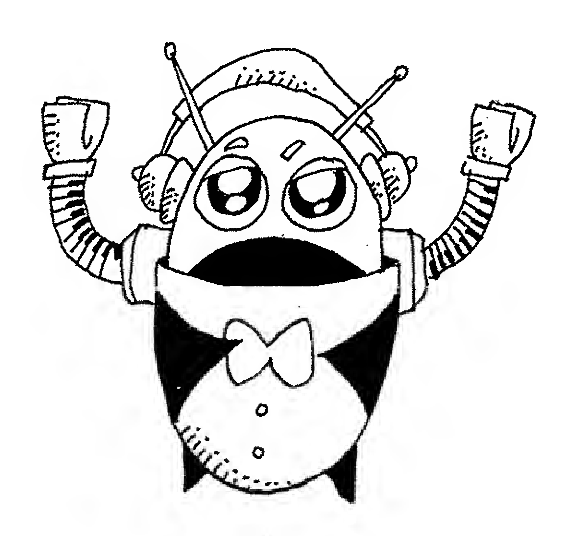
¿Cómo funciona PAINT? Llena de Color una figura. Su formato:



En el SCREEN 2, C1 y C2 deben ser el mismo color.

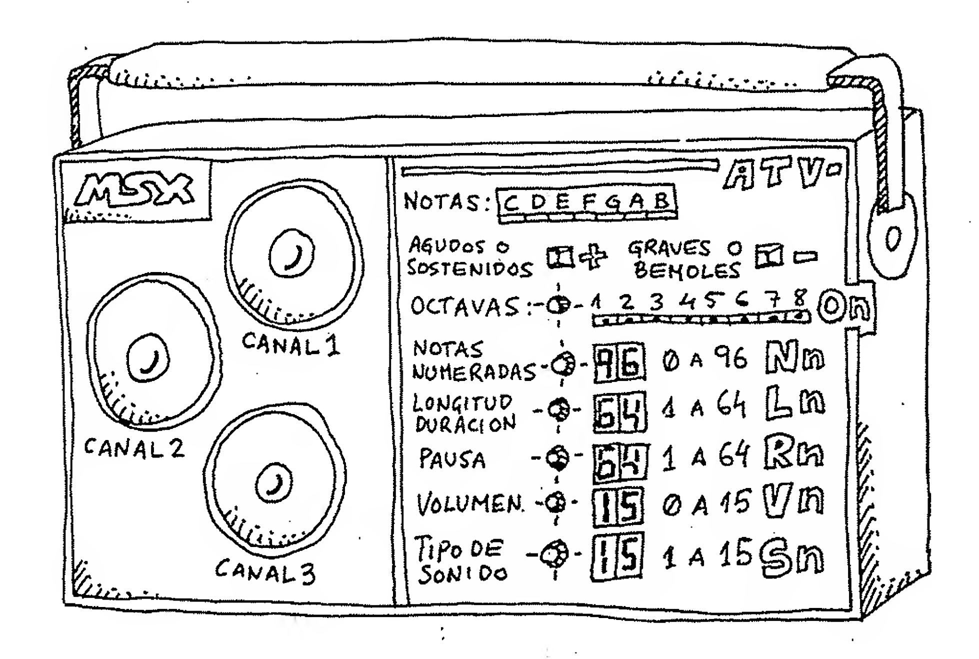
En el SCREEN 3, C1 y C2 pueden ser distintos.

Si la cara es el espejo del alma, la pantalla es la cara del ordenador. Ya le hemos visto la cara a tu MSX
Veamos ahora cómo canta.



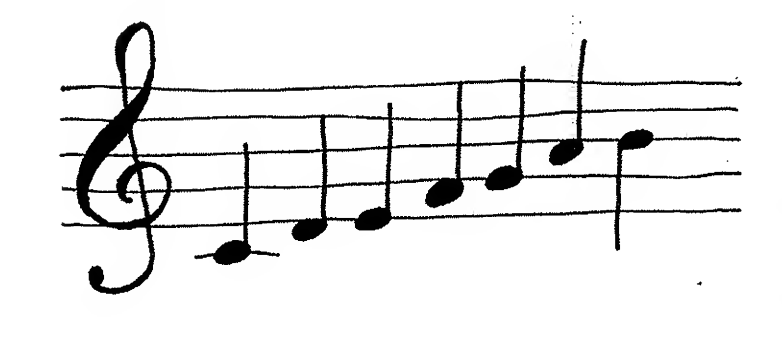
La instrucción musical de tu MSX es PLAY. PLAY al igual que DRAW tiene muchos subcomandos, tiene muchos mandos para regular el volumen, los graves, los agudos, tipo de sonido, duración, octavas, etc.

Es como un estupendo aparato caset lleno de botones.



Eso es PLAY. Lo demás corre de tu cuenta. Si te lo tomas en serio te será fácil aprender solfeo con tu MSX que es un buen profesor.

PLAY "O1S8V7L2C+"





La orquesta funcionará si tú sabes dirigirla.



Y ahora te voy a explicar una función que te va a servir para fabricarte un órgano electrónico, como los que ves en las tiendas. Se trata de INKEY$. Esta función es un controlador del teclado. Recordarás que con la clave KEY podías cambiar el contenido de las teclas de función. Pues exactamente eso hace INKEY$, pero en vez de con las teclas de función, con todo el teclado, excepto **CTRL**, **CTRL STOP**, **SHIFT**. INKEY$ funciona asociada a IF. . . THEN, y este es su formato:


```
10  A$ = INKEY$
20  IF A$ = "1" THEN PLAY "N1"
30  IF A$ = "2" THEN PLAY "N12"
40  IF A$ = "3" THEN PLAY "N24"
50  IF A$ = "4" THEN PLAY "N36"
60  IF A$ = "5" THEN PLAY "N48"
70  IF A$ = "6" THEN PLAY "N60"
80  IF A$ = "7" THEN PLAY "N72"
90  IF A$ = "8" THEN PLAY "N84"
100 IF A$ = "9" THEN PLAY "N96"
110 GOTO 10
```

```
10 A$ = INKEY$
20 IF A$ = "X" THEN TAREA
30 GOTO 10
```

Con INKEY$ se pueden hacer programas divertidísimos. En tus manos está.

Ahora, para aprender, ejecuta los siguientes:

PRACTICAS PROGRAMAS TAREAS

71

```
10 CLS
20 X=RND(-TIME)
30 A=A+1
40 X=RND(1)*50
50 PRINT X
60 IF A<15 THEN GOTO 30
70 END
```

```
10 REM EJEMPLO DE PLAY 1
20 A$="V10S08M1000L8O6F+G+A+O7C+D+C+O
6A+L2A+L8F+G+A+O7C+D+C+O6L2G+L8G+A+O7
C+O6A+G+A+G+G+G+A+O7C+O6A+G+F+L2A+"
100 PLAY A$,B$
```

72

```
10 REM EJEMPLO DE PLAY 2
20 A$="V10S08M1000L8O6F+G+A+O7C+D+C+O
6A+L2A+L8F+G+A+O7C+D+C+O6L2G+L8G+A+O7
C+O6A+G+A+G+G+G+A+O7C+O6A+G+F+L2A+"
30 B$="V05S08M1000L8O2F+G+A+O3C+D+C+O
2A+L2A+L8F+G+A+O3C+D+C+O2L2G+L8G+A+O3
C+O2A+G+A+G+G+G+A+O3C+O2A+G+F+L2A+"
100 PLAY A$,B$
```

73

```
10 REMSONIDO DE LAS NOTAS DEL 1 AL 96
20 CLS
30 INPUT "DAME UN NUMERO ENTRE 1 Y 96
";A
40 IF A<1 OR A>96 THEN 30
50 PLAY "N"+STR$(A)
60 IF PLAY(0) THEN 60
70 GOTO 30
```

74

```
10 REM DIBUJO DE LASZLO MOHOLY-NAGY:
   COMPOSICION K.IV
20 SCREEN 2:COLOR 15,1,1:CLS
30 PI=3.1416
40 DRAW"C14BM60,192E120
50 LINE(112,0)-(119,160),9,BF
60 LINE(115,10)-(120,160),8,BF
70 LINE(121,10)-(124,160),6,BF
80 DRAW"S4C10BM91,160E20D20L20"
90 PAINT(100,158),10
100 CIRCLE(150,102),30,15,1.22*PI,.21
*PI,1.4
110 CIRCLE(155,99),15,6,1.22*PI,.21*P
I,1.4
120 LINE(145,107)-(165,88),6
130 PAINT(155,100),6
140 CIRCLE(94,140),30,4,.22*PI,1.22*P
I,1.4
150 DRAW"C4BM78,159R13E20U16"
160 PAINT(80,150),4
170 GOTO 170
```

75


```
10 REM DIBUJO SUBREALISTA.OBRA DEL
   PROGRAMADOR
20 SCREEN 2,2:COLOR 15,7,6:CLS
30 CIRCLE(120,0),144,11,,,1.4
40 PAINT(120,10),11
50 LINE(0,116)-(255,148),2,BF
60 DRAW"C4BM80,44F12D80G12U104"
70 DRAW"BM183,44G12D80F12U104"
80 DRAW"C14BM126,91H20R46G20L6"
90 PAINT(130,73),14
100 LINE(106,55)-(152,70),15,BF
110 DRAW"C12BM126,116G20R46H20L6"
120 PAINT(130,120),12
130 DRAW"C8BM2,96R80H40G40"
140 PAINT(20,90),8
150 LINE(0,149)-(255,159),12,BF
160 CO$="U16R2U2L6D2R2D16R2"
170 DRAW"C1BM20,116"+CO$
180 DRAW"BM66,116"+CO$
190 DRAW"C12BM10,116F25E3F3E10F5E20
L66"
200 PAINT(20,120),12
210 CIRCLE(230,90),20,5,,,1.4
220 PAINT(230,90),5
230 CIRCLE(236,90),20,4,,,1.3
240 PAINT(230,86),4
250 GOTO 250
```

76

77

```
 10 REM LENGUAJE BASIC
20 SCREEN 2:COLOR 15,1,1:CLS
30 FOR R=1 TO 200
40 X=INT(RND(1)*255):
   Y=INT(RND(1)*192)
50 PSET(X,Y),15
60 NEXT R
70 REM ---------LETRA <B>-------
80 DRAW"BM20,80R20F5D8G3F3D8G5L20U32
   E6R20G6E6F5G5E5D8G9F3E5H3F3D8G5
   BM28,85R10F2D6L12E7G7U8BM28,99R12
   D6G2L10E7G7U7"
90 REM ---------LETRA <A>-------
100 DRAW"BM65,80R20D30L5U10L10D10L5
    U30E5R20G5E5D30G5L5U10L10D10E5U5
    BM70,85R10D8L10E5U3D3R5L5G5U8"
110 REM ---------LETRA <S>-------
120 DRAW"BM110,80R20D5L15D5R15D20L20
    U5R15U10L15U15E5R20G5E5D5G5ER5G6
    E6D20G5L20U5E5R8"
```

```
130 REM ---------LETRA <I>-------
140 DRAW"BM160,80R10D30L10U30E5R10G5
    E5D30G5"
150 REM ---------LETRA <C>-------
160 DRAW"BM200,80R20D5L15D20R15D5L20
    U30E5R20G5E5D5G5L10D15G5E5R15G6E
    6D5G5"
170 GOTO 170
```

```
10 REM FLOR DE CIRCULOS
20 SCREEN 2:COLOR 15,4,4:CLS
30 R=40:PI=4*ATN(1)
40 FOR T=1 TO 360 STEP 20
50 A=PI*T/180
60 X=130+R*SIN(A)
70 Y=80+R*COS(A)
80 CIRCLE(X,Y),R,15,,,1.4
90 NEXT T
100 GOTO 100
```
78

```
10 REM CIRCULOS
20 SCREEN 2:COLOR 15,14,14:CLS
30 H=RND(-TIME)
40 FOR A= 1 TO 50
50 R=RND(1)*20:C=RND(1)*15:S=RND(1)*5
60 X=RND(1)*230+10:Y=RND(1)*170+10
70 CIRCLE(X,Y),R,C,,,S
80 NEXT A
90 GOTO 90
```
79

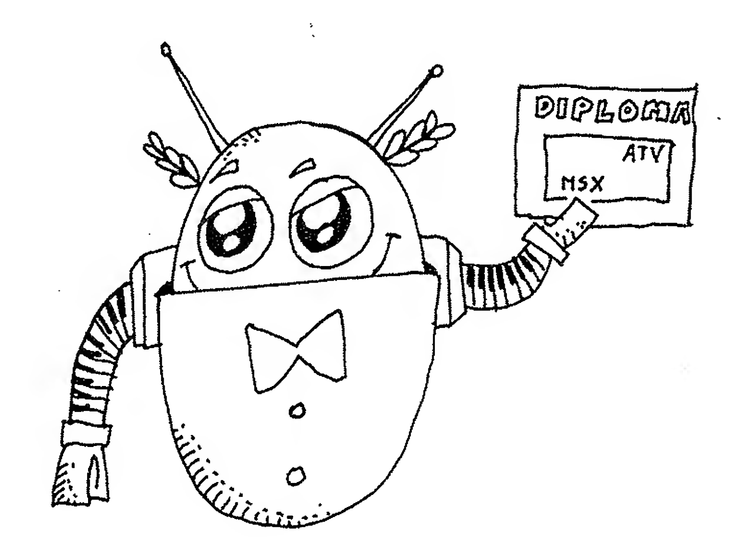
80
```
10 REM PRINCIPIO
20 COLOR15,15,15:SCREEN2
30 X1=4:Y1=4:X=127:Y=96
40 X=X+X1:Y=Y+Y1
50 IF X>255 THEN X1=-4:GOTO40
60 IF X<0 THEN X1=4:GOTO40
70 IF Y>191 THEN Y1=-4:GOTO40
80 IF Y<0 THEN Y1=4:GOTO40
90 C%=1+(14*RND(1))
100 CIRCLE(X,Y),10,C%,,,1.4
110 CIRCLE(X,Y),10,15,,,1.4
120 GOTO40
```

```
10 REM CIRCULOS
20 SCREEN 2:COLOR 15,4,4:CLS
30 FOR R=5E-03 TO 25.5 STEP .2
40 CIRCLE(128,96),50,15,,,R
50 FOR T=1 TO 200:NEXT T
60 NEXT R
70 GOTO 70
80 GOTO 80
```
81

Lección 24

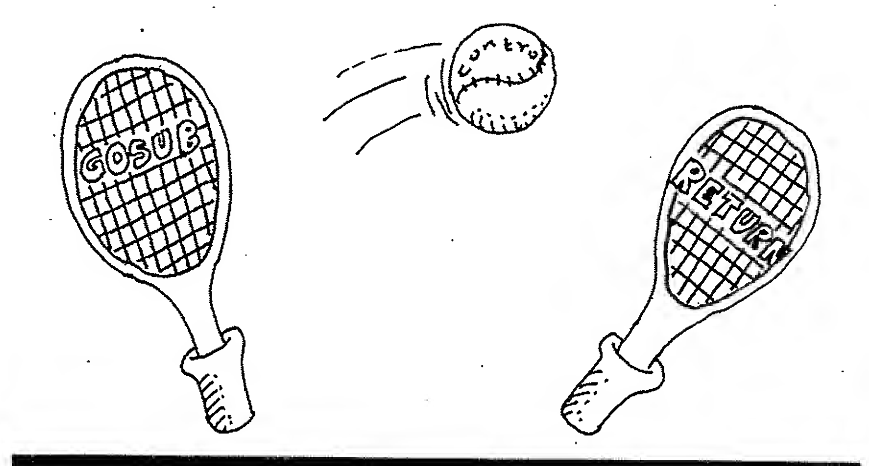
. . . Y vamos con la última.

Eres el mejor compañero de estudios que he tenido. Has soportado con estoica paciencia todos mis humores, y sobre todo has trabajado intensamente para alcanzar un objetivo que, en sí mismo, va a recompensar todos tus esfuerzos : Aprender.



Hoy para terminar, hablaremos de una instrucción muy útil para confeccionar grandes programas, y que por ello, hemos dejado para el final:

GOSUB – RETURN



...REPASA AVANZA...

1. INKEY$. Es un controlador del teclado. Con esta función puedes alterar el valor de las teclas de tu MSX

2. PAINT. Es la pintura de tu MSX. Sirve para dar color a las figuras que se dibujan en SCREEN 2 y SCREEN 3.



La instrucción de hoy es GOSUB. . . RETURN significa : "VE A". . . "REGRESA".

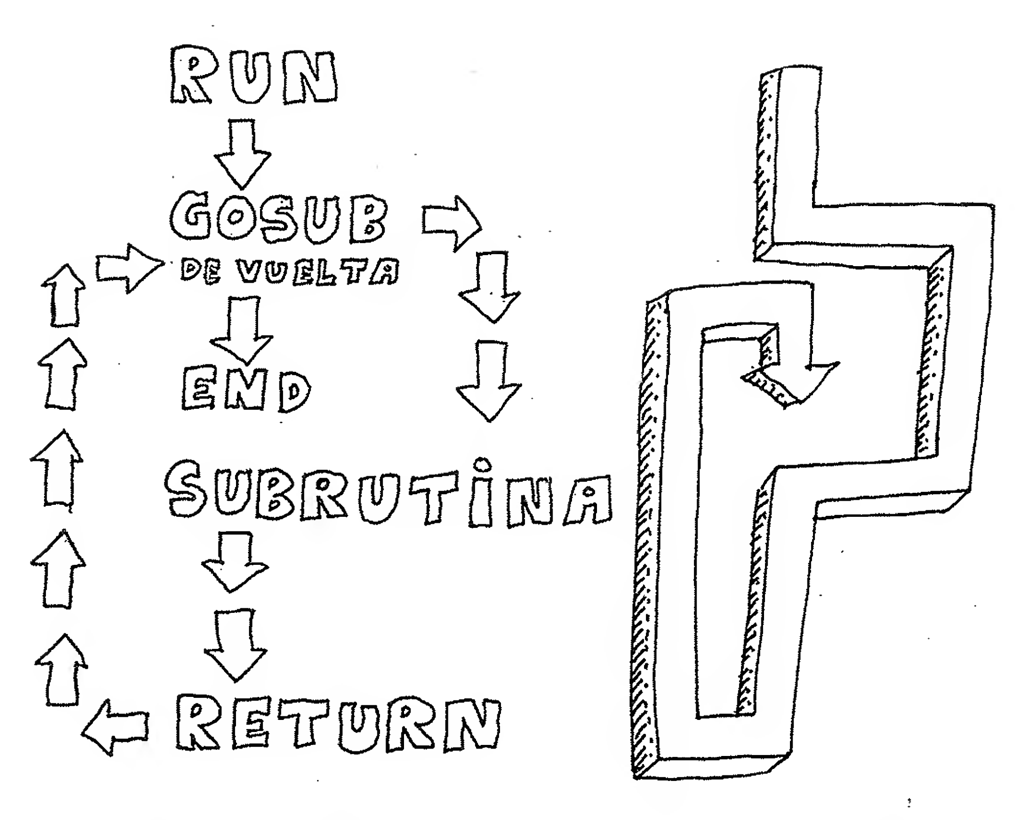
Y para qué sirve. Pues para que el control "VAYA A" una línea determinada, EJECUTE una tarea determinada y RETORNE a la línea de la que partió.

GOSUB da un salto como el que daba GOTO, pero con la diferencia de que luego se encuentra a RETURN, que lo devuelve a la línea de origen, una vez que se ha ejecutado la tarea o SUBRUTINA encomendada.

GOSUB. . . RETURN es la instrucción de las subrutinas.

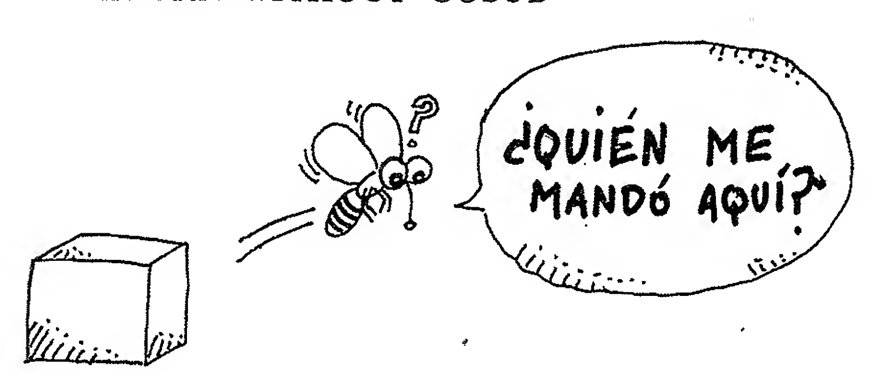
```
10 PRINT "PROGRAMA PRINCIPAL"
20 FOR A = 1 TO 3000 : NEXT A
30 GOSUB 60
40 PRINT "DE VUELTA EN EL PROGRAMA
   PRINCIPAL"
50 END
60 PRINT "SUBRUTINA"
70 FOR B = 1 TO 3000 : NEXT B
80 RETURN
```

La influencia de GOSUB. . . RETURN en el flujo de control es:



Algo que debes tener muy en cuenta es que si olvidas poner GOSUB o lo formulas mal el ordenador te lo advertirá con el mensaje de error:

RETURN WITHOUT GOSUB



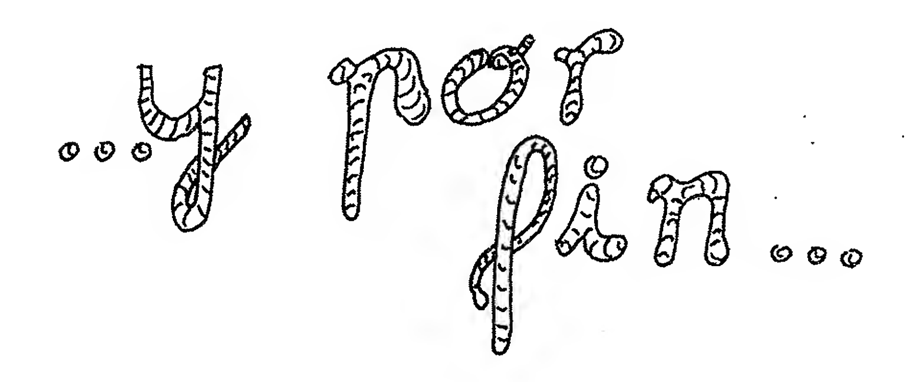
Ejecútalo.

ON ERROR GOTO. Tiene un formato complejo, completado por PRINT ERL, PRINT ERR.

Con PRINT ERL se imprime en la pantalla el número de la línea errónea. Lo has visto en el programa anterior.

Con PRINT ERR se imprime el Código que cada error posee en MSX. (Consulta en el manual de tu MSX los Códigos de Error).

Con PRINT ERL, PRINT ERR el formato es el del programa anterior; sólo tienes que poner una u otra instrucción.

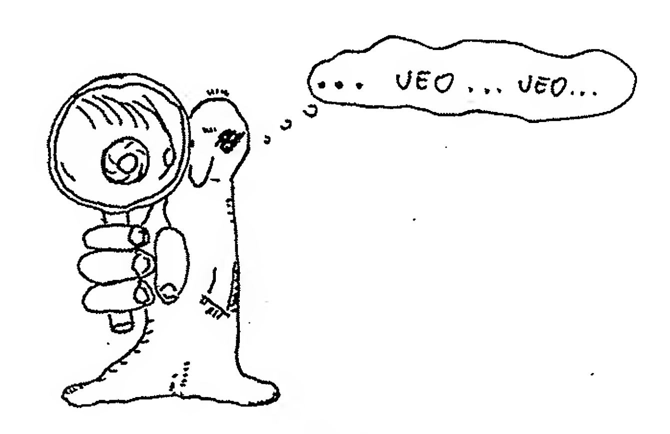


Amigo programador, con esto hemos llegado al final de nuestro curso. Espero sabrás disculpar nuestros defectos, que en ningún momento han sido intencionados. Muchas gracias por tu atención y tu esfuerzo y. . .

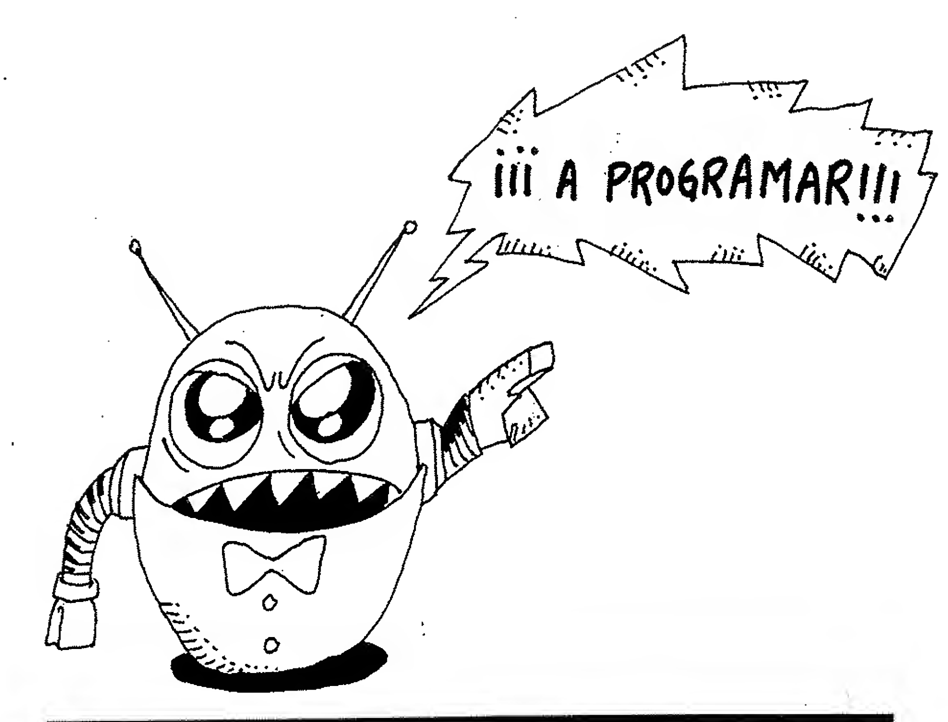
Y hablando de errores. A partir de hoy te vas a enfrentar solo con el ordenador. Cada vez irás haciendo programas más complejos, y con más líneas. En cualquier programa es normal que surjan errores : por olvidos, por teclear mal una instrucción, por uso defectuoso de los periféricos, etc.

Existe en BASIC MSX una maravillosa instrucción que te ayudará a localizar errores dentro de tus programas. Se trata de ON ERROR GOTO.

El propósito de esta instrucción es localizar e identificar un error que se produce en el programa.




```
10 ON ERROR GOTO 50
20 PRINT "AQUI NO HAY ERROR"
30 PRIN "AQUI SI. FALTA LA T"
40 END
50 PRINT ERL
60 RESUME  NEXT
```

PRACTICAS PROGRAMAS TAREAS

82

```
10 REM CALCULO DE LAS FASES DE LA
LUNA
20 GOTO 1000
100 REM INTRODUCCION DE DATOS
105 CLS
110 PRINT TAB(5)"FASES DE LA LUNA"
120 PRINT
140 INPUT "DIA....DD=";D
150 INPUT "MES....MM=";M
160 INPUT "A%O..AAAA=";A
170 IF M<3 THEN M=M+12:A=A-1
180 RETURN
200 REM CALCULOS
210 T=INT(365.25*A)+INT(30.6*(M+1))+D
-694038!
220 T=T/36525!
230 LA=350.737486+1236*T*360
240 LA=LA+307*T+6*T/60
250 LA=LA+51.18*T/3600-5.17*T*T/3600
260 LA=LA-INT(LA/360)*360
```

```
270 LA=INT(LA+.5)
280 RETURN
300 REM PRESENTACION DE RESULTADOS
310 PRINT:PRINT
320 PRINT USING"FASE DE LA LUNA A=###
GRADOS";LA
330 PRINT"PRESIONA UNA TECLA"
340 IF INKEY$="" THEN 340
350 RETURN
1000 REM PROGRAMA PRINCIPAL
1010 GOSUB 100
1020 GOSUB 200
1030 GOSUB 300
1040 GOTO 1010
```

83

```
10 GOSUB 200
20 TIME=0
30 COLOR A%,B%,B%
40 CLS
50 A$=INKEY$
60 IF TIME/50>20 THEN 100
70 IF A$="" THEN 50
80 PRINT A$;
90 GOTO 50
100 A=A+1
110 IF A<10 THEN 10
120 END
200 A%=RND(1)*15
210 B%=RND(1)*15
220 IF A%=B% THEN 200
230 RETURN
```

INSTRUCCION "GOSUB...RETURN"

84

```
10 CLS
20 FOR N%=1 TO 10
30 FOR N=1 TO 10
40 GOSUB 100
50 NEXT N,N%
60 END
100 PRINT N;"*";N%;"=";N*N%
110 FOR M%=1 TO 100:NEXT M%
120 RETURN
```


85

```
10 REM TELEX
20 SCREEN 2:COLOR 15,14,14:CLS
30 OPEN "GRP:" AS 1
40 LINE(20,0)-(20,192),1
50 LINE(230,0)-(230,192),1
60 FOR S=10 TO 190 STEP 10
70 CIRCLE(10,S),3,1,,,1.4
80 CIRCLE(240,S),3,1,,,1.4
90 NEXT S
100 X=30:Y=10:Y1=18:D=18:RESTORE 310:
    GOSUB 200
110 X=30:Y=30:Y1=38:D=16:RESTORE 320:
    GOSUB 200
120 X=30:Y=50:Y1=58:D=16:RESTORE 330:
    GOSUB 200
130 X=30:Y=70:Y1=78:D=4:RESTORE 340:
    GOSUB 200
140 X=30:Y=90:Y1=98:D=5:RESTORE 350:
    GOSUB 200
150 X=30:Y=110:Y1=118:D=5:RESTORE 360
    :GOSUB 200
160 X=30:Y=130:Y1=138:D=8:RESTORE 370
    :GOSUB 200
170 X=30:Y=150:Y1=158:D=4:RESTORE 380
    :GOSUB 200
180 X=30:Y=170:Y1=178:D=7:RESTORE 390
    :GOSUB 200
190 GOTO 190
200 LINE(22,Y1)-(228,Y1+2),1,BF
210 FOR A=1 TO D
220 X=X+10:READ A$
230 PSET(X+10,Y),1:COLOR 1:PRINT #1
    ,";"
240 PSET(X,Y),14:COLOR 14:PRINT #1
    ,";"
250 PSET(X,Y),14:COLOR 1:PRINT #1,A$:
    COLOR 14
260 BEEP
270 NEXT A
280 LINE(22,Y1)-(228,Y1+2),14,BF
290 PSET(X+10,Y),14:COLOR 14:PRINT #1
    ,";"
300 RETURN
310 DATA E,S,T,O, ,E,S, ,U,N, ,E,J,E,
    M,P,L,O
320 DATA D,E, ,L,O, ,Q,U,E, ,P,U,E,D,
    E,S
```

```
330 DATA H,A,C,E,R, ,C,O,N, ,T,U, ,M,
    S,X
340 DATA H,O,L,A
350 DATA A,D,I,E,U
360 DATA A,D,I,O,S
370 DATA S,A,Y,O,N,A,R,A
380 DATA C,H,A,O
390 DATA G,O,O,D,B,Y,E
```

```
10 REM ESTE PROGRAMA EJECUTA UNA NOTA
ENTRE 2 Y 52 PULSANDO LAS LETRAS
MAYUSCULAS
20 REM PULSANDO RETURN SE EJECUTARA
TODO LO PULSADO Y TERMINA EL PROGRAMA
30 CLS
40 A$=INKEY$
50 IF A$="" THEN 40
60 A=ASC(A$):IF A=13 THEN 100
70 GOSUB 180
80 PLAY N$
90 GOTO 40
100 IF PLAY(0) THEN 100
110 CLS
120 FOR M=1 TO LEN(B$)
130 A$=MID$(B$,M,1)
140 GOSUB 200
150 PLAY N$
160 NEXT M
170 END
180 IF A<65 OR A>90 THEN 40
190 B$=B$+A$
200 N=ASC(A$)-65
210 D$=STR$(N*2):N$="N"+RIGHT$(D$,LEN
(D$)-1)
220 PRINT A$;
230 RETURN
```

86

```
10 REM PLAY, PRINT Y GOSUB CON INKEY$
20 A$=INKEY$
30 IF A$="P" THEN GOSUB 60
40 IF A$="R" THEN GOSUB 70
50 GOTO 20
60 PLAY"V10T150O5L2BCGFADEL4BACGDFEL6
   GCFEAFGCDE":PRINT"PLAY":RETURN
70 PRINT "PRINT"
80 RETURN
```

87

88

```
10 REM ESTE PROGRAMA OBTIENE
CUALQUIER TABLA
20 CLS
30 LOCATE 5,3:PRINT "1- SUMAR"
40 LOCATE 5,6:PRINT "2- RESTAR"
50 LOCATE 5,9:PRINT "3- MULTIPLICAR"
60 LOCATE 5,12:PRINT "4- DIVIDIR"
70 LOCATE 5,15:INPUT "NUMERO DE LA TA
BLA";A
80 IF A=1 THEN 200
90 IF A=2 THEN 270
100 IF A=3 THEN 340
110 IF A=4 THEN 410
120 GOTO 20
130 REM OTRA TABLA SI O NO
140 PRINT
150 PRINT "OTRA TABLA ¡S/N¿"
160 A$=INKEY$:IF A$="" THEN 160
170 IF A$="S" THEN 20
180 IF A$<>"N" THEN 160
190 END
200 REM TABLA DE SUMAR
210 CLS
220 INPUT "TABLA DEL NUMERO";N
230 FOR M=1 TO 10
240 PRINT N;"+";M;"=";N+M
250 NEXT M
260 GOTO 140
270 REM TABLA DE RESTAR
280 CLS
290 INPUT "TABLA DEL NUMERO";N
300 FOR M=1 TO 10
310 PRINT N;"-";M;"=";N-M
320 NEXT M
330 GOTO 140
340 REM TABLA DE MULTUPLICAR
350 CLS
360 INPUT "TABLA DEL NUMERO";N
370 FOR M=1 TO 10
380 PRINT N;"*";M;"=";N*M
390 NEXT M
400 GOTO 140
410 REM TABLA DE DIVIDIR
420 CLS
430 INPUT "TABLA DEL NUMERO";N
440 FOR M=1 TO 10
450 PRINT N;"/";M;"=";N/M
460 NEXT M
470 GOTO 140
```

89

```
10 REM HELICOPTERO
20 SCREEN 2,2:COLOR 15,4,4:CLS
30 RESTORE 390:S=1:GOSUB 330
40 RESTORE 400:S=2:GOSUB 330
50 RESTORE 410:S=3:GOSUB 330
60 LINE(0,190)-(80,130),7
70 LINE(80,130)-(180,170),7
80 LINE(180,170)-(200,150),7
90 LINE(200,150)-(255,190),7
100 PAINT(125,180),7
110 SOUND 0,2
120 SOUND 1,5
130 SOUND 3,8
140 SOUND 6,2
150 SOUND 7,7
160 SOUND 8,10
170 SOUND 9,10
180 SOUND 11,1
190 SOUND 12,1
200 SOUND 13,15
210 SOUND 8,30
220 SOUND 9,20
230 FOR M=20 TO 15 STEP -1
240 X=X+2:P=P+1
250 PUT SPRITE 0,(X,40),15,1
260 PUT SPRITE 1,(X+15,40),15,2
270 PUT SPRITE 4,(P,30),15,3
280 SOUND 8,M
290 SOUND 9,M
300 NEXT M
310 IF X>510 THEN X=0
320 GOTO 230
330 SP$=""
340 FOR K=1 TO 32
350 READ A:SP$=SP$+CHR$(A)
360 NEXT K
370 SPRITE$(S)=SP$
380 RETURN
390 DATA 15,0,0,0,64,96,96,127,127,0,
    0,0,0,0,0,0,255,0,0,1,7,59,59,251
    ,250,13,3,2,34,31,0,0
400 DATA 255,64,64,240,200,196,194,
    242,242,250,252,32,34,252,0,0,254
    ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
410 DATA 0,0,0,1,6,8,16,32,32,32,16,
    12,3,0,0,0,0,0,0,128,96,24,6,1,1,
    6,8,8,48,192,0,0
```


90

```
10 REM REBOTE DE PELOTA
20 SCREEN 2:COLOR 15,4,4:CLS
30 LINE(10,10)-(240,180),15,B
40 GOSUB 120
50 A=2:B=2:Y=10:X=10
60 X=X+A
70 Y=Y+B
80 IF X<12 OR X>232 THEN A=-A
90 IF Y<12 OR Y>172 THEN B=-B
100 PUT SPRITE 0,(X,Y),15,0
110 GOTO 60
120 DATA 24,60,126,126,60,24,0,0
130 RESTORE 120:S$=""
140 FOR K%=1 TO 8
150 READ A:S$=S$+CHR$(A)
160 NEXT K%
170 SPRITE$(0)=S$
180 RETURN
```

91

```
10 REM MOVIMIENTO ESTELAR
20 SCREEN 2:COLOR 15,4,4:CLS
30 DATA 0,0,0,24,24,0,0,0
40 RESTORE 30:S$=""
50 FOR K%=1 TO 8
60 READ A:S$=S$+CHR$(A)
70 NEXT K%
80 SPRITE$(0)=S$
90 X=120:X1=120:Y=90:Y1=90:A=2
100 IF X>220 OR X<40 THEN A=A*-1
110 X=X+A:X1=X1-A:Y=Y+A:Y1=Y1-A
120 PUT SPRITE 0,(X,Y),15,0
130 PUT SPRITE 1,(X,Y1),15,0
140 PUT SPRITE 2,(X1,Y),15,0
150 PUT SPRITE 3,(X1,Y1),15,0
160 PUT SPRITE 4,(X,90),15,0
170 PUT SPRITE 5,(X1,90),15,0
180 PUT SPRITE 6,(120,Y),15,0
190 PUT SPRITE 7,(120,Y1),15,0
200 GOTO 100
```

92

```
10 REM TIOVIVO
20 SCREEN 2,3:COLOR 15,4,4:CLS
30 LINE(120,60)-(125,170),7,BF
40 LINE(55,170)-(185,175),6,BF
50 LINE(40,175)-(205,185),11,BF
60 LINE(40,60)-(205,55),11,BF
70 LINE(40,55)-(122,10),2
80 LINE(122,10)-(205,55),2
90 LINE(205,55)-(40,55),2
100 DRAW"C6BM122,9U10F5G5"
110 PAINT(123,5),6
120 PAINT(120,30),2
130 LINE(60,62)-(61,170),1,BF
140 LINE(100,62)-(101,170),1,BF
150 LINE(180,62)-(181,170),1,BF
160 LINE(145,62)-(146,170),1,BF
170 LINE(1,185)-(255,185),14
180 LINE(122,12)-(122,54),15
190 LINE(122,12)-(60,54),15
200 LINE(122,12)-(185,54),15
210 LINE(40,55)-(122,10),15
220 LINE(122,10)-(205,55),15
230 GOSUB 410
240 S$="V10T20004L2C03L4B04CE05L1C04
    L4B05CDC04B05C04E05C04L1BR1"
250 T$="V8T22002L4E04ER64E02E04ER64E
    02E04ER64E02E04ER64E02E04ER64E02
    E04ER64E02E04ER64E"
260 PLAY S$,T$
270 FOR Y=60 TO 140
280 PUT SPRITE 0,(131,Y),10,1
290 PUT SPRITE 1,(86,200-Y),2,1
300 PUT SPRITE 2,(166,200-Y),6,1
310 PUT SPRITE 3,(46,Y),7,1
320 NEXT Y
330 FOR Y=140 TO 60 STEP -1
340 PUT SPRITE 0,(46,Y),7,1
350 PUT SPRITE 1,(86,200-Y),2,1
360 PUT SPRITE 2,(166,200-Y),6,1
370 PUT SPRITE 3,(131,Y),10,1
380 NEXT Y
390 GOTO 270
400 DATA 0,0,0,0,238,138,138,170,170,
    170,238,0,0,0,0,0,0,0,0,0,238,74,
    74,74,74,74,78,0,0,0,0,0
410 RESTORE 400
420 S$=""
430 FOR K%=1 TO 32
440 READ A:S$=S$+CHR$(A)
450 NEXT K%
460 SPRITE$(1)=S$
470 RETURN
```


93

```
10 REM HOLA EN AVION
20 SCREEN 2,1:COLOR 15,12,7:CLS
30 RESTORE 460:S=1:GOSUB 510
40 RESTORE 470:S=5:GOSUB 510
50 RESTORE 480:S=4:GOSUB 510
60 RESTORE 490:S=3:GOSUB 510
70 RESTORE 500:S=2:GOSUB 510
80 PI=3.1416
90 DRAW"C7BM0,100M80,40M120,70M160,30
   M255,110"
100 PAINT(1,1),7
110 LINE(0,180)-(255,180),2
120 PAINT(1,190),2
130 CIRCLE(30,30),20,10
140 PAINT(30,30),10
150 NUBE$="R4FR3F3EUERER7FRR3D2G2D2R2
    FDFD4GLGLGL6HL2D3G2L10HLHLH2GLGL4
    HL3H2LUHU5HU3EUERER2ER7FRER3"
160 DRAW"C15BM200,10"+NUBE$
170 PAINT(205,15),15
180 DRAW"BM 77,1"+NUBE$
190 PAINT(80,10),15
200 DRAW"BM 115,25"+NUBE$
210 PAINT(120,33),15
220 X=30:GOSUB 410
230 X=60:GOSUB 410
240 X=100:GOSUB 410
250 X=140:GOSUB 410
260 X=190:GOSUB 410
270 X=220:GOSUB 410
280 X=110:GOSUB 410
290 FOR X=1 TO 680
300 PUT SPRITE 1,(X,40),14
310 PUT SPRITE 2,(X-20,24),4
320 PUT SPRITE 3,(X-40,24),4
330 PUT SPRITE 4,(X-60,24),4
340 PUT SPRITE 5,(X-80,24),4
350 NEXT X
360 S=2:P=X-20:GOSUB 580
370 S=3:P=X-40:GOSUB 580
380 S=4:P=X-60:GOSUB 580
390 S=5:P=X-80:GOSUB 580
400 GOTO 400
410 LINE(X,180)-(X,120),8
420 CIRCLE(X,130),15,2,0,1*PI
430 CIRCLE(X,125),10,2,1.9*PI,1.1*PI
    ,2
440 CIRCLE(X,125),10,2,1.9*PI,1.1*PI
    ,1
```

EVALUCION FINAL

PROGRAMADORES DE HASTA 12 AÑOS. Código secreto. Tienes que hacer un programa para que al pulsar cualquier tecla, se imprima otra distinta. Es decir, tienes que hacer un código secreto, que sólo tú conocerás. Este código debe activarse al teclear tú una palabra clave, y se desactivará al pulsar otra distinta.

PROGRAMADORES DE HASTA 16 AÑOS. El ordenador te pedirá un número. Una vez que se lo hayas dado, debe decirte todas las combinaciones de tres números que den como resultado el anterior. Por ejemplo: el ordenador te da el número 6. Las combinaciones posibles de tres números que sumen 6 son:

Ø Ø 6
Ø 1 5
Ø 2 4
Ø 3 3
etc.

PROGRAMADORES MAYORES DE 17 AÑOS. Tienes que hacer un frontón con, al menos, una raqueta y una pelota, con marcadores electrónicos, de tiempo, tanteo, etc.

Independientemente del grupo de edad al que pertenezcas, podrás mandarnos el supuesto de un grupo distinto al tuyo, si bien esto se tendrá en cuenta a la hora de evaluar tu ejercicio.

Evaluación: ....

Nivel: ....

Número: ....

Nombre: ....

1.er Apellido: ....

2.º Apellido: ....

Edad: ....

Dirección: ....

....

Ciudad: ....

LISTADO DEL PROGRAMA

COMENTARIO DEL PROGRAMA

Solo con tus manos. INFORMATICA-BASIC ATV pone a tu alcance hacer un viaje a través del maravilloso universo de la Informática y el BASIC-MSX. Tu ordenador es algo más que un juguete: es una herramienta; la Informática es una poderosa herramienta que tú puedes aprender a manejar con nosotros. No queremos enseñarte a matar marcianitos. . . vamos a enseñarte a programar.

¿Quieres aprender INFORMATICA-BASIC-MSX con nosotros?

